Міністерство освіти і науки України
Дніпровський національний університет
залізничного транспорту імені академіка В. Лазаряна

КОСОЛАПОВ А.А., ЖУКОВИЦЬКИЙ І.В.

КОНЦЕПТУАЛЬНЕ ПРОЕКТУВАННЯ КОМП'ЮТЕРНИХ СИСТЕМ РЕАЛЬНОГО ЧАСУ

(МОДЕЛІ, МЕТОДИ І АЛГОРИТМИ)

Дніпро – 2018

УДК 004.94:519.875.5
Концептуальне проектування комп’ютерних систем реального часу (моделі, методи і алгоритми) . Дніпро: ДНУЗТ, 2018. – 274 с.
ISBN XXXXXXXXXXXX

Рецензенти:

Ткачов В.В., доктор технічних наук, професор, завідувач кафедри автоматизації та комп’ютерних систем Національного технічного університету «**Дніпровська політехніка**»

Хандецький В.С., доктор технічних наук, професор, завідувач кафедри електронних обчислювальних машин Дніпровського національного університету імені О. Гончара

В даній науковій монографії вперше запропонований комплекс моделей, методів і методик вирішення завдань концептуального проектування комп'ютерних систем, що працюють в реальному масштабі часу. Особливу увагу приділено ресурсозберігаючим методам формування раціональних технічних структур розподілених інформаційно-керуючих систем (ІКС). Застосування запропонованих математичних засобів ілюструється прикладами системного проектування реальних ІКС автоматизації процесів на залізничних сортувальних станціях. Багато технічних рішень і програми автоматизації розрахунків характеристик ІКС на ранніх стадіях їх проектування захищені авторськими свідоцтвами.

Монографія буде корисною як молодим дослідникам (студентам старших курсів, магістрам і аспірантам), так і фахівцям, пов'язаним з розробками складних ІКС в різних галузях промисловості і на транспорті.

Рекомендовано до друку
Вченою радою Дніпровського національного університету
залізничного транспорту ім. акад. В Лазаряна,
протокол від 5.11.2018 року № 5

Адреса редакції:
49010, м. Дніпро, вул. Лазаряна, 2
e-mail: XXXX@xxx.ua, +380 xxxxxxxxxx

Ministry of Education and Science of Ukraine
Dniprovsky National University of
Railway Transport named after Academician V. Lazaryan

KOSOLAPOV A. A., ZHUKOVITSKYI I.V.

CONCEPTUAL DESIGN
OF REAL TIME COMPUTER SYSTEMS
(MODELS, METHODS AND ALGORITHMS)

Dnipro - 2018

UDC 004.94:519.875.5
Conceptual Design of Real Time Computer Systems (Models, Methods, and Algorithms) . Dnipro: DNURT, 2018. – 274 c.
ISBN XXXXXXXXXXXX

Reviewers:
Tkachev V.V., Dr. Sc. Eng., Professor of National Technical Universityy **«Dniprovska Polytechnic»**
Khandetsky V.S., Dr. Sc. Eng., Professor 0f Dnipro National University named after O. Honchar

In this scientific monograph a complex of models, methods and techniques for solving problems of conceptual design of computer systems, which operating in real time scale is proposed for the first time. Particular attention is paid to resource-saving methods for the development of rational technical structures of distributed information-control systems (ICS). The application of the proposed mathematical tools illustrates the examples of the system design of real-time ICS automation processes at railway sorting stations. Many technical solutions and automation software for calculating the characteristics of ICS in the early stages of their design are protected by copyright certificates.

The monograph will be useful as young researchers (senior students, masters and graduate students), as well as professionals involved in the development of complex ICS in various industries and in transport.

The monograpf was approved
at a meeting of the DNURT Academic Council
5. 11.2018, the protocol No. 5

Editorial Address:
49010, Lazarjan street, 2, Dnipro,.
e-mail: XXXX@xxx.ua, +380 xxxxxxxxxx

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ 8

ПЕРЕДМОВА 11

РОЗДІЛ 1

АНАЛІЗ НАПРЯМКІВ І ОСНОВНИХ ЗАВДАНЬ РОЗВИТКУ ІНФОРМАЦІЙНИХ СИСТЕМ (ІС) 13

1.1 Розвиток інфрасрукиури інформатизації 13

1.2 Зміна парадигми комп'ютеризації 14

1.3 Архітектура комп'ютерних систем 16

1.4 Загальні тенденції і етапи удосконалення архітектури ІС 19

1.5. Розвиток архітектури систем автоматизації (на прикладах АСУ сортувальних станцій) 28

1.5.1 Чотирирівнева структура АСК ВП УЗ-Є 30

1.5.2 Обчислювальні і комунікаційні ресурси системи 33

1.5.3 Особливості побудови і функціонування 34

1.5.4 Граничні проектні оцінки характеристик системи 35

1.5.5 Оцінка експлуатаційних показників АСК ВП УЗ-Є 38

РОЗДІЛ 2

ТЕОРЕТИЧНІ ОСНОВИ ДЕКОМПОЗИЦІЇ СИСТЕМ АВТОМАТИЗАЦІЇ ТЕХНОЛОГІЧНИХ ПРОЦЕСІВ 41

2.1 Особливості проектування складно структурованих систем керування 41

2.2 Розробка багатосортних алгебраїчних моделей дискретних систем керування реального часу 49

2.2.1 Час як змінна 50

2.2.2 Події і принципи декомпозиції дискретних систем керування сортувальними гірками 52

2.2.3 Основні визначення дискретних систем реального часу 58

2.2.4 Композиція і декомпозиція структур 59

2.2.4.1 Послідовна композиція 60

2.2.4.2 Паралельна композиція 63

2.2.4.3 Композиція зі зворотнім зв'язком 64

2.2.5 Багатосортна алгебра, морфізм і гомоморфізм структур дискретних систем керування 66

РОЗДІЛ 3

МЕТОДИ ПОБУДОВИ І ДОСЛІДЖЕННЯ ІНФОРМАЦІЙНИХ, ФУНКЦІОНАЛЬНИХ І ТЕХНІЧНИХ СТРУКТУР ДЕЦЕНТРАЛІЗОВАНИХ АВТОМАТИЗОВАНИХ СИСТЕМ КЕРУВАННЯ 71

3.1 Принципи побудови М-моделей розподілених інформаційно-керуючих систем 71

3.2 Завдання структурної оптимізації М-моделі і загальний алгоритм його вирішення 76

3.3 Принципи застосування М-моделей в процесі проектування інформаційно-плануючої системи сортувальної станції 79

3.4 Методи пошуку раціональних варіантів структур інформаційних систем на основі генетичних алгоритмів 83

3.4.1 Метод раціонального розподілу інформаційних потоків в інформаційно-керуючих системах 83

3.4.2 Метод пошуку раціональних «зіркоподібних» структур інформаційних систем з використанням кодування Прюфера 100

3.5 Ресурсоощадні методи вибору технічних структур децентралізованих інформаційно-керуючих систем залізничних сортувальних станцій 110

3.5.1 Ресурсоощадний метод дослідження умов доцільності децентралізації функцій керування 110

3.5.2 Метод вибору технічних структур цифрових керуючих систем за критерієм ефективного використання обчислювальних ресурсів 120

3.5.3 Ресурсоощадний підхід до оцінювання витрат на побудову цифрових розподілених керуючих систем 135

РОЗДІЛ 4

МОДЕЛІІ ТА МЕТОДИ РОЗРАХУНКУ ХАРАКТЕРИСТИК КОМП'ЮТЕРНИХ СИСТЕМ КЕРУВАННЯ РЕАЛЬНОГО ЧАСУ 140

4.1 φ-транзакція як основна модель для оцінки інформаційних і часових характеристик сервіс-орієнтованих систем 140

4.1.1 Визначення поняття «φ-транзакція» 140

4.1.2 Метод оцінки характеристик системи керування реального часу в процесі проектування 143

4.2 Метод обробки φ-транзакцій і розрахунку їх характеристик 151

4.3 Методика, аналітичні моделі та інструментальні засоби вибору системи пріоритетів обробки φ-транзакцій 153

4.4 Методи аналізу надійності систем з використанням теорії розмитих множин 161

4.4.1 Базові положення теорії для оцінки показників надійності 161
4.4.2 Аналіз надійності нечітких систем на основі розмитих множин 167
4.4.3 Метод використання різних видів розмитих множин для оцінки надійності нечітких систем 173
4.4.4 Приклади рішення задач оцінки надійності нечітких систем без резервування і з багатократним резервом 179
4.5 Інформаційно-часові діаграми для опису алгоритмів роботи децентралізованих інформаційних систем реального масштабу часу 186
4.6 Логіко-лінгвістична система керування сповільнювачем прицільної гальмівної позиції на сортувальній гірці 194
4.7 Принципи нечіткої маршрутизації для розподілених систем і мереж 198
4.8 Онтологічні моделі в задачах автоматизації сортувальних станцій 203
4.9 Інтелектуальні методи ідентифікації зібраних маршрутів в парках сортувальної станції 211
4.9.1. Основні положення 211
4.9.2. Використання додаткових сигналів від наземного устаткування 213
4.9.3 Алгоритм підвищення ймовірності ідентифікації зібраних маршрутів 213
4.9.4 Приклад реалізації алгоритму 216
4.9.5 Ефективність запропонованого алгоритму 217
4.10 Синтез програмного забезпечення інформаційно-керуючих систем реального часу на основі теорії абстрактних автоматів 218
4.10.1. Проблеми автоматизації введення інформації в інформаційно-керуючі системи залізничного транспорту 218
4.10.2. Проектування автоматної моделі 220
4.10.3. Синтез програмного продукту 224
РОЗДІЛ 5
КОМПЛЕКСНИЙ ПІДХІД ДО КОНЦЕПТУАЛЬНОГО ПРОЕКТУВАННЯ І ВДОСКОНАЛЕННЯ ІНФОРМАЦІЙНИХ СИСТЕМ РЕАЛЬНОГО МАСШТАБУ ЧАСУ 230
5.1 Процеси проектування інформаційних систем 230
5.2 Принципи і загальні вимоги до методології проектування систем реального масштабу часу 233
5.3 Методики концептуального (системного) проектування складних ІС 235
5.4 Науково-методичний комплекс системного інтегратора (КСІ) 241
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ 248

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

АСК	автоматизована система керування
АСК ВП УЗ-Є	єдина АСК вантажними перевезеннями "Укрзалізниці"
АСК МР	АСК маршрутами руху на сортувальній станції (рос. АСУ МД)
АСК РСГ	АСК розформування составів на гірці (рос. АСУ РСГ)
АСК СС	АСК сортувальної станції (рос. АСУ СС)
АСОТУСС	автоматизована система організаційно-технологічного управляння сортувальної станції
АС УВП	аналітичні сервери управління вантажними перевезеннями
АРШ	система автоматизованого керування швидкістю відчепів на гірці
АП	абсолютні пріоритети
БД	база даних
БП	безпріоритетна дисципліна обслуговування заявок
ВКХС	контроль відчепу і вимір ходових властивостей
ВП	відносні пріоритети
ДСРМЧ	дискретні СРМЧ
ЕОМ	електронна обчислювальна машина
ЗП	змішані пріоритети
ІАРШ	автоматичне регулювання швидкості відчепів в інтервальних гальмівних позиціях
ІКК СС	інформаційно-керуючий комплекс сортувальної станції (рос. ИУК СС)
ІКОС	інформаційно-керуюча обчислювальна система
ІПС СС	інформаційно-плануюча система сортувальної станції
КеОК	керуючий обчислювальний комплекс (рос. УВК)
КеОМ	керуюча обчислювальна машина (рос. УВМ)
КПТЗА	комплекс програмно-технічних засобів автоматизації
КСІ	комплекс системного інтегратора

КТЗ	комплекс технічних засобів
ЛЗ	лінгвістичне забезпечення
МЗ	математичне забезпечення
МКЕОМ	мікро-ЕОМ
МКК	мікроконтролер
МККпс	мікроконтролерна підсистема
МКП	мікропроцесор
НО	напольне (наземне) обладнання
НА	низова автоматика
ОЗ	організаційне забезпечення
ОС	операційна система
ПАРШ	автоматичне регулювання швидкості відчепів в прицільних гальмівних позиціях
ПЗ	програмне забезпечення
ПЗО	пристрої зв'язку з об'єктом (рос. УСО)
РІКОС	розподілена ІКОС
СМО	система масового обслуговування
СРМЧ	система реального масштабу часу
ТЗ	технічне завдання
ТОА	технологічний об'єкт автоматизації
ТОК	технологічний об'єкт керування
УІП	умовна інформаційна потужність АСК
ФПБ	функціональний програмний блок
ФЧД	функціонально-часові діаграми
ЦКС	цифрова керуюча система
Capacity reserves	продуктивність, пропускна здатність, ємність
HC-FS	Hard Client - File Server (Товстий клієнт - Файловий сервер)

HC-DBS	Hard Client - Data Base Server (Товстий клієнт - Сервер баз даних)
p2p-архітектура	peer to peer архітектура; однорангова, децентралізована або пірінгова мережа
rsДСРМЧ	реактивна ДСРМЧ
SC-AS-DBS	Slim Client - Application Server - Data Base Server (Тонкий клієнт - Сервер додатків - Сервер баз даних)
SWC-WS-DBS	Slim Web Client - Web Server - Data Base Server (Тонкий веб-клієнт - Веб-сервер - Сервер баз даних)
SWC-WS-AS-DBS	Slim Web Client - Web Server - Application Server - Data Base Server (Тонкий веб-клієнт - Веб-сервер - Сервер додатків - Сервер баз даних)
“The bottom line”	верхні витрати, повні проектні вартості
Timeliness	своєчасність

ПЕРЕДМОВА

Дана книга містить результати багаторічних теоретичних досліджень і розробок авторів, пов'язаних з проектуванням автоматизованих систем управління складними виробничими процесами для залізниць України та ближнього зарубіжжя від перших мікропроцесорних АСУ ТП на промислових комп'ютерах СМ 1800 до сучасної інтегрованої системи управління вантажними перевезеннями на базі IBM R780 (АСК ВП УЗ). Особливості цих систем: вони відносяться до класу складних систем за умовною інформаційною потужністю, за характером і кількістю виконуваних завдань, за своєю структурою, яка істотно впливає на їх ефективність; вони спираються на мережені технології взаємодії підсистем. Практичний досвід розробки перерахованих систем підтвердив відомі оцінки вітчизняних і закордонних вчених та фахівців, що говорять про важливість початкових стадій створення систем, які в цілому називають концептуальним або системним проектуванням. У європейських стандартах також використовується назва «System Definition». Відповідно до вітчизняних стандартів сюди відносяться розробка концепції побудови автоматизованої системи, розробка технічного завдання на систему, ескізний і технічний проекти. Усунення можливих помилок в проекті, допущених на цих стадіях, вартує при впровадженні та експлуатації системи на підприємстві на два порядки дорожче їх виправлення на ранніх етапах. Це свідчить про важливість повного і якісного концептуального проектування. Для складних систем забезпечення якості проектно-дослідницьких робіт неможливо без застосування математичних моделей, методів, інженерних методик і засобів автоматизації розрахунків їх характеристик.

У даній книзі розглядаються розроблені авторами і перевірені на практиці математичні моделі, методи, алгоритми, методики концептуального проектування з прикладами розв'язання ряду завдань в

процесі створення складних інформаційно-керуючих систем для АСУ залізничними сортувальними станціями.

В цій монографії основна увага приділяється аналітичним моделям і методам. Такий підхід призводить до досить простих рішень, і в той же час є досить результативним, дозволяючи в компактній аналітичній формі визначити всі найважливіші характеристики систем.

Серйозні труднощі виникали в зв'язку з відсутністю стійкої термінології з обробки даних. Для цієї книги термінологія відбиралася і визначалася відповідно до найбільш авторитетних публікацій. При трактуванні термінів автори хотіли досягти в першу чергу однозначності в розумінні читачем матеріалів, що викладається, і забезпечити термінам певну ємність, необхідну для широкого застосування теоретичних результатів.

Книга розрахована на широке коло наукових і інженерно-технічних працівників, які займаються розробкою інформаційно-керуючих систем, на спеціалістів, які вирішують завдання удосконалення структури автоматизованих систем. Книга буде корисною аспірантам, магістрам і студентам старших курсів, які цікавляться питаннями проектування територіально і функціонально розподілених комп'ютерних систем і мереж.

Автори

РОЗДІЛ 1
АНАЛІЗ НАПРЯМКІВ І ОСНОВНИХ ЗАВДАНЬ РОЗВИТКУ ІНФОРМАЦІЙНИХ СИСТЕМ (ІС)

1.1 Розвиток інфраструктури інформатизації

В даний час у всіх галузях економіки, у тому числі і на транспорті, відбуваються кардинальні зміни в області інформаційних технологій і систем [142]. Так, відповідно до Транспортної стратегії України до 2020 року основними напрямками її реалізації є забезпечення доступності та якості транспортних послуг за рахунок створення «комплексних інформаційних систем управління, контролю та ідентифікації вантажів і контейнерів...», при цьому серед пріоритетів розвитку видів транспорту називається «створення автоматизованої системи управління залізничними перевезеннями через головний та регіональні центри управління, централізація управління рухом поїздів» та «забезпечення розвитку опорних сортувальних станцій» [13].

На мережі залізниць України розташовано 36 сортувальних станцій, з них 15 є вирішальними, при цьому тільки 4 з них автоматизовані (частково). Вирішення поставлених ІТ-фахівцям галузі завдань вимагає конструктивної оцінки стану і проблем в області інформатизації галузі. В даний час можна виділити наступні тенденції, які потребують особливої уваги: розширення наборів вирішуваних завдань на існуючій комп'ютерній базі; вимоги формування повідомлень та видачі керуючих дій в реальному масштабі часу; збільшення кількості інформаційних систем різного призначення; розвиток «засобів і технологій інформатизації» [33]; масштабність, комплексний характер, інтеграція і ускладнення систем і процесів їх проектування та модернізації; інтелектуалізація завдань, що вирішуються; відсутність єдиної системи понять і визначень у галузі прикладних ІТ-технологій і систем, що призводить до несумісності

багатьох рішень, до їх дублювання та нераціонального використання фінансових ресурсів; слаба технічна оснащеність систем [33] (зараз, наприклад, термін служби серверів баз даних і серверів додатків становить 3 роки [157], на сортувальних станціях зношеність релейного і комп'ютерного обладнання становить від 50 до 90% [8]); застаріле програмне забезпечення, недостатньо розвинута мережа передачі даних [33] (при підключенні клієнтів через мережу ІНТЕРНЕТ, необхідно враховувати, що збільшення трафіку становить не менше ніж 300% на рік [157]); недостатня типізація пристроїв, велика різноманітність систем, відсутність сховищ даних, ручне введення інформації [33] – все це можна віднести до однієї проблеми - відсутність методики системного проектування складних інформаційних систем (за оцінками фахівців Microsoft 54% IT-проектів в США зазнали невдачі через ігнорування опрацювання архітектурних питань; за останні два-три роки час на розробку архітектури інформаційних систем збільшився з 10-15 до 50% від загального часу виконання проекту [2]; відзначимо, що розробка архітектури систем виконується на ранніх стадіях проектування - розробка концепції побудови системи , ТЗ, ескізний і технічний проект).

Викладені труднощі, пов'язані з розробкою, модернізацією і впровадженням сучасних систем автоматизації, говорять про актуальність пошуку науково-обґрунтованих відповідей на проблемні питання.

1.2 Зміна парадигми комп'ютеризації

Розвиток засобів обчислювальної техніки та інформаційних технологій останнім часом призвело до зміни парадигм у розумінні сутності та основних вимог до систем. Нагадаємо, що парадигма (з грецької «приклад, модель, зразок») - це сукупність фундаментальних наукових установок, уявлень і термінів, що приймається та розділяється науковим співтовариством і об'єднує більшість його членів. Вона

забезпечує спадкоємність розвитку науки і наукової творчості. Фундаментальне уявлення про інформаційні системи можна звести до ієрархії основних вимог до них (див. рис. 1.1).

На ранніх етапах розвитку ЕОМ та систем основною вимогою користувачів було забезпечення максимальної швидкості ОБЧИСЛЕНЬ або швидкодії при вирішенні завдань вимірювання, контролю та управління. Тому всі вели розмови про ОБЧИСЛЮВАЛЬНІ системи і системи автоматизації автоматизовані системи управдіння підприємствами (АСУП), технологічними процессами (АСУТП) і т.п. Питання інтерфейсу і доступу до даних були на другому плані.

На початку нового століття, яке стали називати інформаційним, мережа ІНТЕРНЕТ стала не тільки транспортним засобом, але й великим інформаційним ресурсом, з'явилися WEB-системи та відбулася зміна парадигми і ієрархії вимог: заговорили про ІНФОРМАЦІЙНІ СИСТЕМИ, які об'єднують всі типи комп'ютерних систем незалежно від розв'язуваних задач. У таких системах основною вимогою є доступ до інформації «тут і зараз». Мережеві технології надали можливість роботи в реальному масштабі часу для територіально-розподілених систем.

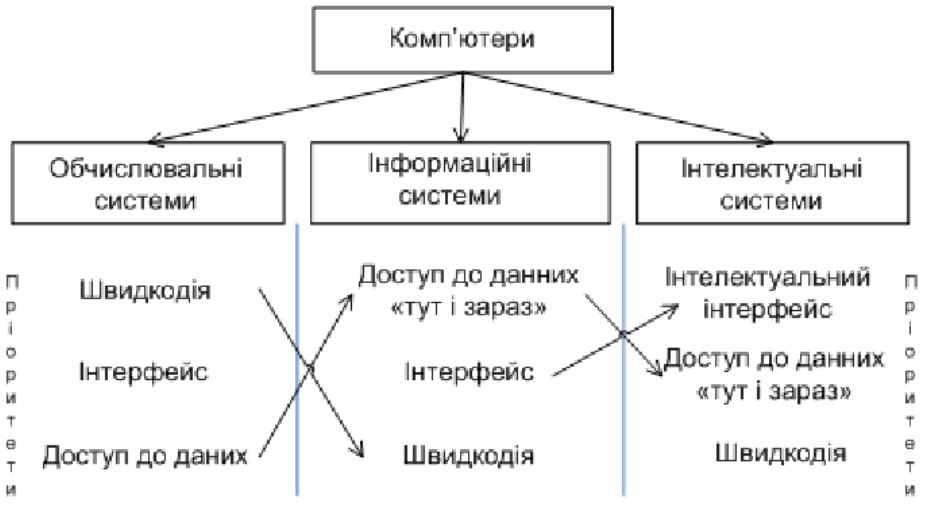


Рис.1.1 Зміна парадигми комп'ютерних систем

Поява нової парадигми спостерігається в наш час, коли користувачів не турбує швидкодія систем, доступ до даних, але проявляється зростаюча

потреба в якості очікуємих даних, які володіли б релевантністю, достовірністю та допомагали б у процесі прийняття рішень. У даному випадку мова йде про інтелектуальний інтерфейс, ідея якого була вперше запропонована японцями в проекті стратегічних обчислювальних машин п'ятого покоління в 1981 році. Така машина була побудована на базі моделей і методів штучного інтелекту, в якості базової мови використовувався Пролог. Нова система була в одному примірнику виготовлена в 1991 році, але виявилася дуже дорогою і незатребуваною на початку 90-х. І тільки в наш час почали активно проводитись роботи по створенню комп'ютерних систем з елементами штучного інтелекту, або інтелектуальних систем. У новій парадигмі на перший план виходять питання інтелектуалізації інтерфейсу і вирішення прикладних задач з використанням моделей і методів штучного інтелекту. Найбільш поширеними системами нового типу стали інтелектуальні транспортні системи (ІТС) [17].

Перспектива вирішення багатьох проблем - у переході до інтелектуального залізничного транспорту, що поєднує взаємодію «розумного» локомотива і «розумної» станції. Розробкою ІТС займаються провідні світові фірми. Створення та впровадження таких систем підтримуються міжнародними транспортними організаціями. Інтелектуальні технічні засоби дозволяють полегшити роботу персоналу, забезпечити логічний контроль за його діями в штатних і позаштатних ситуаціях. З їх допомогою є можливість проводити розширену і оперативну діагностику роботи устаткування і приймати рішення щодо забезпечення надійності, безпеки і життєздатності процесу перевезень [17].

1.3 Архітектура комп'ютерних систем

Спочатку необхідно визначити поняття «архітектура» як базове поняття будь-якої системи.

Системна архітектура, за стандартом ANSI / IEEE 1471-2000is, - це

«фундаментальна організація системи, реалізована в її компонентах, зв'язки цих компонентів один з одним і зовнішнім середовищем та принципах, що визначають структуру і розвиток системи». Фактично в цьому визначенні мова йде про сукупність різних структур.

Це підтверджує і аналіз матеріалів різних джерел, виконаний у [45]: термін «архітектура системи» часто є синонімом терміна «структура системи». При використанні терміну «архітектура системи» на перший план висувається складний, багатоаспектний характер структури системи. Ось деякі найбільш відомі варіанти визначення терміна (поняття) «архітектура системи» (див. таблиця 1.1).

Вітчизняні стандарти і нормативні документи [4] не використовують термін «архітектура системи». Але в них визначаються:

- види структур ІС - функціональна, технічна, організаційна, програмна, інформаційна;

- основні структурні компоненти ІС - користувачі і комплекс засобів автоматизації (КЗА);

- види забезпечення ІС - програмне, інформаційне, технічне, організаційне, методичне, математичне, лінгвістичне, правове та ін.;

- необхідність виділення структури функціональних систем і підсистем ІС, опису складу та характеристик автоматизованих функцій і завдань ІС.

Фактично, будь-яка архітектура ІС, незалежно від парадигми, - це множина взаємопов'язаних структур, яка описується наступним виразом:

$$АІКС = МЕТА \cap (КТЗ \cup ПЗ \cup МЗ \cup ІЗ \cup ЛЗ \cup ОЗ \cup МетрЗ \cup ДЗ) \quad (1.1)$$

де КТЗ або ТЗ (технічне забезпечення) - комплекс технічних засобів системи; ПЗ - програмне забезпечення (загальне і спеціальне ПЗ); МЗ - математичне забезпечення (сукупність математичних моделей, методів і алгоритмів); ІЗ - інформаційне забезпечення (опис сигналів, принципів класифікації і кодування інформації, опис масивів, форм, нормативно-

Таблиця 1.1

Визначення поняття архітектура інформаційної системи

Джерело	Визначення
IEEE Recommended Practice for Architectural Description, Draft 3.0 of IEEE P1471, May 1998	Архітектура - високорівнева концепція системи, яка враховує оточення
ISO-15704, Industrial automation systems – Requirements for enterprise-reference architectures and methodologies. August 20, 1999	Архітектура системи - опис (модель) основного розташування та взаємозв'язків частин системи (фізичного або концептуального об'єкта або суті)
ANSI/IEEE Std 1471-2000, Recommended Practice for Architectural Description of Software-Intensive Systems	Архітектура - фундаментальна організація системи, вміщена в своїх компонентах, в їх взаємовідносинах, в оточенні, а також **принципи, що визначають проектування, створення і розвиток системи**
Міжнародний центр наукової та технічної інформації» - http://www.icsti.su	Архітектура - концепція, яка визначає структуру і взаємозв'язок компонентів складного об'єкта
Словник http://www.glossary.ru	Архітектура інформаційної системи - концепція, визначальна модель, структура, функції і взаємозв'язок компонентів інформаційної системи

довідкової та інших видів інформації); ЛЗ - лінгвістичне забезпечення (сукупність мовних засобів спілкування персоналу з системою); ОЗ - організаційне забезпечення (організаційна структура та інструкції оперативному персоналу); МетрЗ - метрологічне забезпечення (засоби забезпечення заданих достовірних характеристик вимірювальних функцій системи); ДЗ - документаційне забезпечення системи. Мета - мета створення системи. Всі ці види забезпечення характеризуються набором взаємопов'язаних статичних і динамічних структур, які формуються в процесі проектування системи та об'єднані загальною концептуальною

схемою для досягнення цілей створення при мінімізації сумарних витрат [78, 140].

Залежно від парадигми домінує та чи інша частина в даному складі видів забезпечення. Для *обчислювальних систем* - це ТЗ (швидкодія), ПЗ (організація обчислень), *для інформаційних систем* - ІЗ (бази даних), ПЗ (мережеве програмне забезпечення), ТЗ (мережеві структури), *для інтелектуальних систем* - МЗ (моделі та методи представлення та обробки знань), ІЗ (бази знань), ЛЗ (мови інтелектуального спілкування), ПЗ (мови логічного і функціонального програмування, оболонки експертних систем), ОЗ (користувачі системи), ТЗ (спеціальні пристрої введення-виведення).

Запропоновану багатоланкову формулу архітектури ІС можна порівняти з поїздом, в якому є локомотив, який забезпечує рух до поставленої мети - це КТЗ. Ми не можемо «робити більш важкими» види забезпечень у «поїзді» без узгодження з можливостями КТЗ. Наприклад, використання складних алгоритмів управління швидкістю скочування відчепів у гальмівних позиціях на основі моделювання фізичних процесів руху відчепів на гірці обмежуються швидкодією контролерів, що використовуються. Нарощування кількості завдань у діючих системах або збільшення кількості клієнтів, що потребують обслуговування, небезпечно без урахування потрібних обчислювальних ресурсів і пропускної спроможності каналів. Ігнорування таких оцінок призводить до відмов систем при критичних навантаженнях або до зниження якості приймаємих рішень. Тому вибір «локомотива» для підтримки відповідного «составу» видів забезпечення є важливим завданням, як при проектуванні, так і при модернізації інтелектуальних систем.

1.4 Загальні тенденції і етапи удосконалення архітектури ІС

Архітектуру інформаційної системи автоматизованого керування можна розглядати як модель, яка визначає інформаційно-функціональну організацію системи (1.1).

Компоненти інформаційної системи за функціями, які вона виконує, можна розділити на три шари: шар відображення, шар бізнес-логіки і шар доступу до даних (див. рис. 1.2).

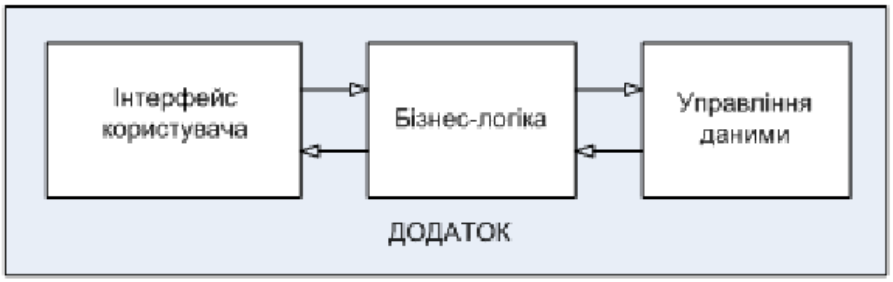


Рис. 1.2 Структура додатків інформаційних систем

Шар відображення (інтерфейсу) - все, що пов'язано із взаємодією з користувачем: натискання кнопок, рух миші, малювання зображення, виведення результатів пошуку, тощо.

Бізнес-логіка - правила, функції, алгоритми реакції додатка на дії користувача або на внутрішні події, правила та програми обробки даних.

Шар доступу до даних - зберігання, вибірка, модифікація і видалення даних, пов'язаних з розв'язуваної додатком прикладною задачею.

З точки зору програмно-апаратної реалізації компонентів інформаційної системи можна виділити ряд типових архітектур ІС, які удосконалювалися з розвитком мережевих технологій. Можна виділити п'ять основних етапів розвитку (див. рис. 1.3).

Спочатку на сортувальних станціях використовувались багато термінальні системи з розподілом часу на базі молодших моделей ЄС ЕОМ - ЄС 1010, ЄС 1022, а на сортувальних станціях - системи реального масштабу часу на базі міні-ЕОМ СМ-2М. З появою мікропроцесорних ЕОМ і локальних мереж спочатку будувалися однорангові мережі для обміну файлами, в яких всі комп'ютери - рівноправні, а потім виникла ідея виділення окремої машини для зберігання загальнодоступних файлів. Це були перші 2-х рівневі системи "товстий клієнт - файл-сервер" (НС-FS).

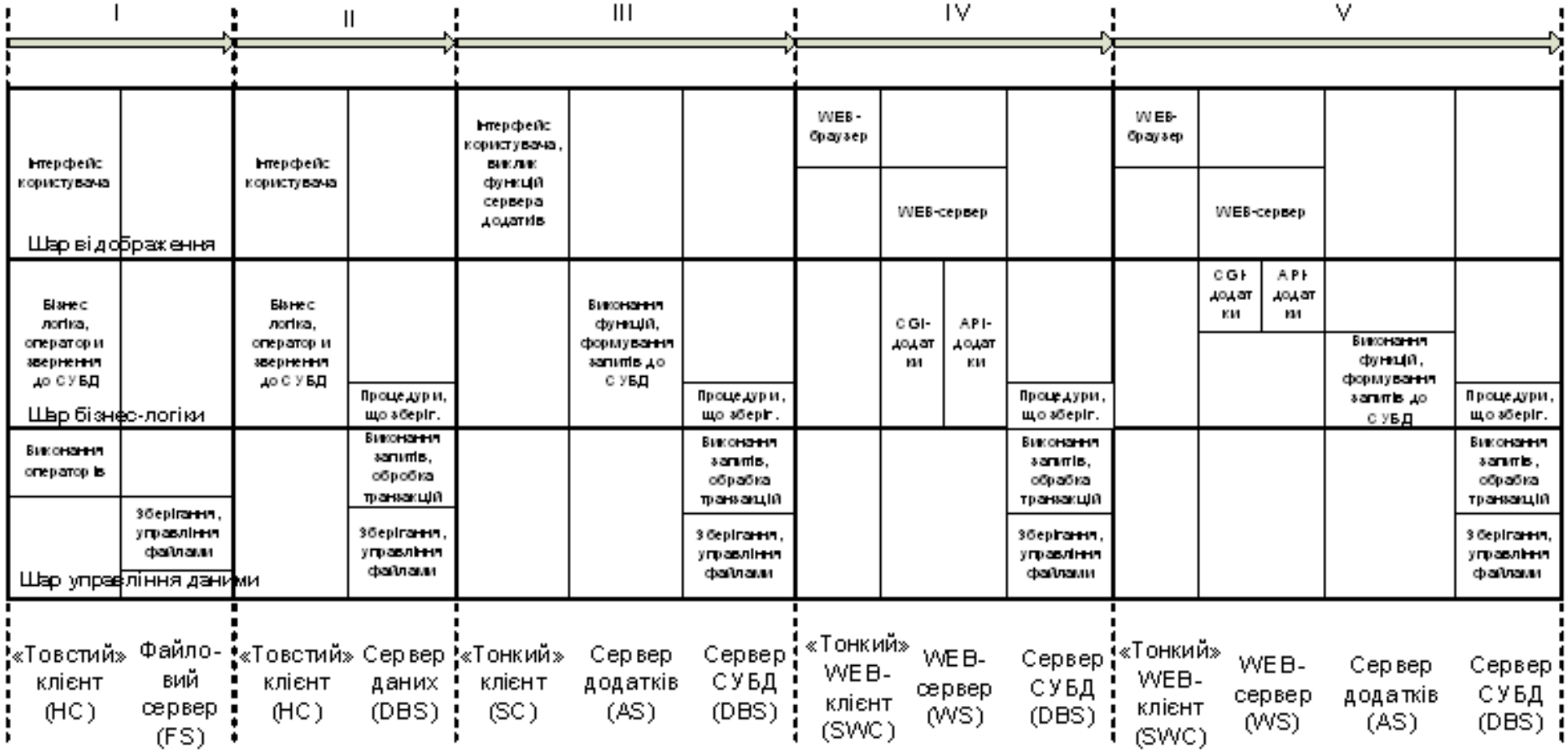


Рис. І.3 Етапи розвитку архітектури

Файл-серверні додатки - додатки, схожі за своєю структурою з локальними додатками і використовують мережевий ресурс для зберігання програм і даних. Функції сервера: зберігання даних і програм. Функції клієнта: обробка даних (виключно на стороні клієнта). Кількість клієнтів обмежена десятками. Перевагами таких систем стали: багатокористувацький режим роботи з даними; зручність централізованого управління доступом і низька вартість розробки. У той же час виявилися й недоліки: низька продуктивність і надійність, слабкі можливості розширення.

Недоліки архітектури з файловим сервером випливають головним чином з того, що дані зберігаються в одному місці, а обробляються в іншому. Це означає, що їх потрібно передавати по мережі, що призводить до дуже високих навантажень на мережу і, внаслідок цього, різкого зниження продуктивності додатків при збільшенні числа одночасно працюючих клієнтів. Другим важливим недоліком такої архітектури є децентралізоване вирішення проблем цілісності і узгодженості даних і одночасного доступу до даних. Таке рішення знижує надійність додатків.

Наступний етап розвитку архітектури пов'язаний з розвитком систем управління базами даних (СУБД) і абстрагуванням від внутрішнього представлення даних (фізичної схеми даних). Тепер клієнтські програми маніпулюють даними на рівні логічної схеми. Використання архітектури "товстий клієнт-сервер БД" (HC-DBS) дозволило створювати надійні (в сенсі цілісності даних) багатокористувацькі ІС з централізованою базою даних, незалежні від апаратної (а часто і програмної) частини сервера БД і підтримуючі графічний інтерфейс користувача на клієнтських станціях , пов'язаних локальною мережею. При цьому витрати на розробку додатків суттєво скорочувалися.

Відзначимо головні особливості цієї архітектури. Клієнтська програма працює з даними через запити до серверного ПЗ. Базові функції додатка розділені між клієнтом і сервером.

Нова архітектура дозволила забезпечити повну підтримку багатокористувацької роботи та гарантію цілісності даних. Одночасно стали наявні її недоліки:

а) бізнес-логіка додатків залишилася в клієнті, тобто вони залишалися "товстими"; при будь-якій зміні алгоритмів, треба оновлювати ПО користувача на кожному клієнтському місці;

в) високі вимоги до пропускної здатності комунікаційних каналів з сервером, що перешкоджає використання клієнтських станцій інакше як в локальній мережі;

с) слабкий захист даних від злому, особливо від недобросовісних користувачів системи;

d) висока складність адміністрування і налаштування робочих місць користувачів системи;

е) необхідність використовувати потужні ПК на клієнтських місцях ("товсті" клієнти);

f) висока складність розробки системи через необхідність використання бізнес-логіки та забезпечення інтерфейсу користувача в одній програмі.

Неважко помітити, що більшість недоліків класичної або 2-х шарової архітектури клієнт-сервер виникає від використання клієнтської станції в якості виконавця бізнес-логіки ІС. Тому логічним кроком подальшої еволюції архітектури ІС з'явилася ідея "тонкого клієнта", тобто розділення алгоритмів обробки даних на частини, пов'язані з відображенням інформації в зручному для людини уявленні, з первинною перевіркою даних (клієнтська частина) і пов'язані з виконанням бізнес-функцій і функцій роботи з базами даних (серверна частина). Це так звана проміжна 2,5-шарова архітектура. Використання збережених процедур і функцій обробки даних бізнес-логіки на стороні сервера скорочують трафік, збільшують безпеку.

Отже така організація системи вельми нагадує організацію перших

багато термінальних систем з тією лише різницею, що на місці користувача стоїть не термінал, а персональний комп'ютер з графічним інтерфейсом. Звичайно, таке повернення до майже унітарної системи сталося вже на іншому технологічному рівні. Обов'язковим стало використання СУБД з усіма їх перевагами. Програми для серверної частини пишуть, в основному, на спеціалізованих мовах, користуючись механізмом збережених процедур сервера БД. Таким чином, на рівні логічної організації, ІС в архітектурі клієнт-сервер з тонким клієнтом розщеплюється на три шари - шар даних, шар бізнес-функцій (збережені процедури) і шар відображення. Нажаль, зазвичай, у такій схемі побудови ІС не вдається написати всю бізнес-логіку додатку на не призначених для цього вбудованих мовах СУБД. Тому, дуже часто деякі бізнес-функції реалізуються в клієнтській частині систем, яка від цього невідворотно "товстішає". Тому, що фізично такі ІС складаються з двох компонентів, цю архітектуру часто називають 2,5-шаровий клієнт-сервер.

На відміну від 2-х шарової архітектури 2,5-шарова архітектура зазвичай не вимагає наявності високошвидкісних каналів зв'язку між клієнтською і серверною частинами системи, так як по мережі передаються вже готові результати обчислень - майже всі обчислення проводяться на серверній стороні. Істотно поліпшується також і захист інформації - користувачам даються права на доступ до функцій системи, а не на доступ до її даних і т.д. Однак разом з перевагами унітарного підходу архітектура 2,5 переймає і всі його недоліки, якось: обмежену масштабованість, залежність від програмної платформи, обмежене використання мережевих обчислювальних ресурсів. Крім того, програми для серверної частини системи пишуться на вбудованих в СУБД мовах опису збережених процедур, призначених для валідації даних і побудови нескладних звітів, а зовсім не для написання ІС масштабу підприємства. Все це знижує швидкодію системи, підвищує трудомісткість створення та модифікації ІС і самим негативним чином позначається на вартості апаратних засобів, необхідних для її функціонування.

Для вирішення цих проблем і була запропонована так звана 3-х шарова архітектура клієнт-сервер (SC-AS-DBS). Основною її відмінністю від архітектури 2,5 є фізичний поділ програм, що відповідають за зберігання даних (СУБД) від програм, що ці дані обробляють (сервер додатків, application server (AS)). Такий поділ програмних компонентів дозволяє оптимізувати навантаження як на мережеве, так і на обчислювальне устаткування комплексу.

Компоненти три ланкової архітектури SWC-WS-DBS, з точки зору програмного забезпечення, реалізують певні браузери, web-сервера і сервера БД. Місце будь-якого з цих компонентів може зайняти програмне забезпечення будь-якого виробника. Наприклад, сервер БД може бути представлений MySQL-сервером або MS SQL-сервером; сервер додатків технологіями: ADO.NET, ASP.NET і web-сервером IIS або Apache; роль клієнта виконує будь-який web-браузер.

Переваги тришарової архітектури: a) тонкий клієнт; b) між клієнтською програмою і сервером додатків передається лише мінімально необхідний потік даних - аргументи для функцій, що викликаються і значення цих функцій, що повертаються; c) сервер додатків може бути запущений в одному або декількох екземплярах на одному або декількох комп'ютерах, що дозволяє ефективно і безпечно використовувати обчислювальні потужності організації; d) дешевий трафік між сервером додатків і СУБД; трафік між сервером додатків і СУБД може бути більшим, однак це завжди трафік локальної мережі, а їх пропускна здатність досить велика і дешева. В крайньому випадку, завжди можна запустити AS і DBS на одній машині, що автоматично зведе мережевий трафік до нуля; e) зниження навантаження на DBS в порівнянні з 2,5-шарової схемою, а значить і підвищення швидкості роботи системи в цілому; f) дешевше нарощувати функціональність і оновлювати ПО.

Перехід до 4-хуровневой архітектурі SWC-WS-AS-DBS стався з розвитком ІНТЕРНЕТ-технологій, коли швидкості передачі даних в мережі

стали тотожні зі швидкостями виконання багатьох додатків та істотно збільшилося навантаження на WEB-сервер обслуговуванням запитів, що надходять. В цей же час (з 2010 рр.) з'являються технології віртуалізації обчислень і перші центри обробки даних (ЦОД) - фабрики, що надають інформаційні послуги (сервіси) з налаштованими обчислювальними ресурсами для зберігання і обробки даних і необхідним для цього програмним забезпеченням для віддалених користувачів в мережі. Вперше ідея ЦОДів була сформульована в середині 80-х років минулого сторіччя фірмою IBM, коли в період повального захоплення серверами, персональними комп'ютерами і мережевими структурами вони відзначали, що в мережі не обійтися без потужних і високонадійних центрів на базі мейнфреймів. І ввели термін мережецентричної архітектури мереж і мережецентричних обчислень. В цей же час 1981-1991 роки в Японії був розроблений стратегічний комп'ютер 5-го покоління з інтелектуальним інтерфейсом, який був побудований на мережах мікропроцесорів з потужною базою знань. Це фактично ЦОД, що з'явився на 30 років раніше свого часу. Закінчену в Японії розробку закрили і визнали отримані рішення неефективними. І ось через 30-років, завдяки високошвидкісному ІНТЕРНЕТ, ця ідея втілилася в ЦОДах.

На рис. 1.4 наведено варіант реалізації двох рівнів AS-DBS в системі АСК ВП УЗ [38]. На цих рівнях показані: один із способів збільшення продуктивності системи (кластеризація AS і DBS), використання мереж з різними швидкостями передачі даних і організація резервування.

Зазначені етапи розвитку архітектури інформаційних систем залежно від розвитку інформаційних технологій та пропускної здатності мереж передачі даних доводять, що складаються сприятливі умови для інтеграції автоматизованих систем різних рівнів. При цьому вкрай актуальною стає уніфікація інтеграційних рішень, так як це дозволяє скорочувати терміни робіт і знижувати витрати практично на всіх етапах життєвого циклу автоматизованих систем, включаючи розробку, впровадження та експлуатацію [55].

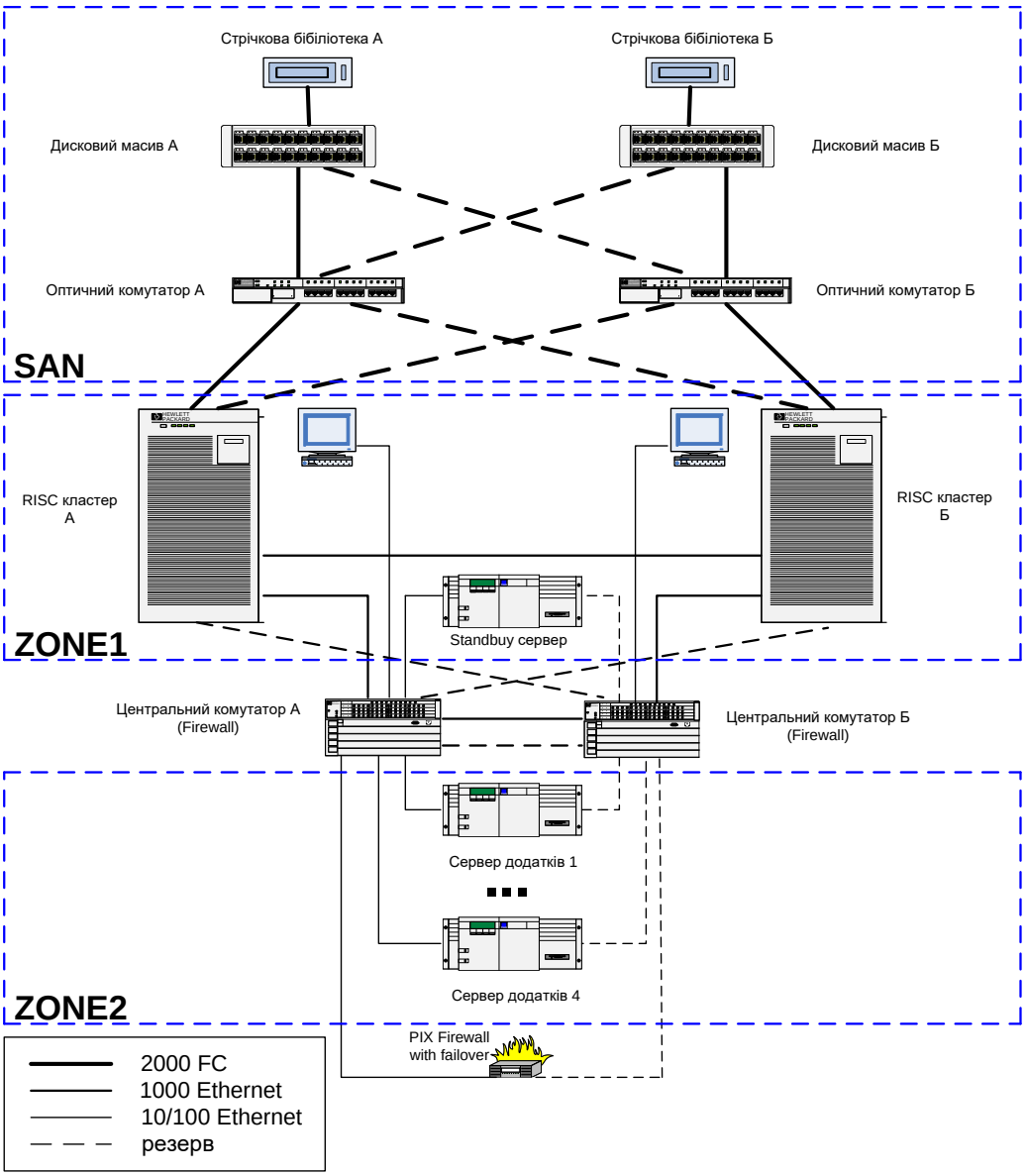


Рис. 1.4 Варіант технічної структури (рівні AS-DBS) АСК ВП УЗ

У Росії в 2004 році в Концепції побудови архітектури АСК РЗД визначені базові поняття, загальні вимоги та підходи до проектування бізнес-архітектури та системної архітектури.

Обов'язкові для виконання вимоги до архітектури вперше були нормативно закріплені розпорядженням № 1272р від 21.12.2005 року "Про затвердження загальних вимог до інтеграції і взаємодії автоматизованих систем". Вони передбачали, зокрема, що ІС повинні мати три ланкову архітектуру, а прикладні функції реалізовані на серверах додатків і доступні через їх програмний інтерфейс. Інтеграція систем повинна

здійснюватися шляхом взаємодії їх серверів додатків, прямий доступ до баз даних не допускається. Необхідна побудова загальної корпоративної шини взаємодії на основі протоколів, що мають статус міжнародних, і сервіс-орієнтованої архітектури (SOA) [55].

Така організація взаємодії та інтеграції систем дозволяє казати про їх інтероперабельність, тобто сумісність на рівні єдиних протоколів інформаційного обміну [118].

ВИЗНАЧЕННЯ. Інтероперабельність (англ. interoperability - здатність до взаємодії) - це спроможність системи, інтерфейси якої повністю відкриті, взаємодіяти і функціонувати з іншими системами без яких-небудь обмежень доступу і реалізації [159].

Як зазначається в [55], для підвищення технологічності побудови загальної моделі інтеграції АСК РЗД доцільно використовувати готові інструменти проектування бізнес-процесів в інтеграції із засобами моделювання систем і з Реєстром автоматизованих систем.

1.5 Розвиток архітектури систем автоматизації (на прикладах АСУ сортувальних станцій)

Автоматизація сортувальних станцій починалася в рамках комплексної програми автоматизації залізничного транспорту (АСКЗТ) на основі електронних обчислювальних машин ЄС ЕОМ. В кінці 1970-х - на початку 1980-х років з'являються перші моделі, що відображають формування, рух і розформування потягів. Паралельно з'являється модель сортувальної станції - основа автоматизованої системи управління сортувальної станцією і перша система АСУ СС (1975 рік). В системі вперше була реалізована видача плану формування ТГНЛ та сортувального листка на основі автоматизованого оформлення ТГНЛ і бази даних призначень плану формування. Особливо слід відзначити серед автоматизованих функцій ведення моделі накопичення вагонів на коліях

сортувального парку, формування составу по накопиченню вагонів, видачу на сформований поїзд натурного листа і довідки до маршруту машиніста. Перші АСК СС будувалися на ЄС 1010 - молодшої з моделей ЄС ЕОМ. В Україні системи були запущені в експлуатацію на станціях Дарниця та Куп'янськ-Сортувальний. Надалі система модернізувалася шляхом заміни ЄС 1010 на ЄС 1022.

Розвиток програмно-технічних засобів, пов'язаний з впровадженням потужних моделей фірми IBM, активізував створення вагонних моделей та моделей поїздів мережевого рівня і в 1980-х роках з'являється автоматизована система оперативного управління перевезеннями (АСОУП) і на її основі - системи автоматизованого диспетчерського центру управління (АДЦУ) . АСОУП, як система дорожнього рівня, забезпечила автоматизацію інформаційного обміну між АСУ СС станцій.

З утворенням незалежних держав роботи, пов'язані з автоматизацією сортувальних станцій, в Росії, Білорусі та Україні пішли по своїми незалежними напрямками. У Росії розробляються системи АСК СТ [11], АІСТ [137], АСТРА СС [130], в Білорусі - система АГАТ [50], в Україні - з'являється Комплексна система електронного обміну документами (КСЕОД-СС), розроблена фахівцями ІОЦ Південної залізниці дороги за участю інженерів програмістів Миронова І.М., Хатунцева К.В., а також фахівців ІОЦ Львівської та Південно-Західної залізниць. Функціонально система КСЕОД-СС була еквівалентна АСК СС, але завдяки новим технічним засобам - персональним комп'ютерам - була більш ефективною за швидкодією, інтерфейсом, енергоспоживанням. Систему можна було розгортати на серверах будь-яких рівнів. Маючи широкі функціональні можливості система КСЕОД-СС була рекомендована для впровадження на залізницях України і досі працює на деяких вирішальних станціях.

З появою системи АСК ВП УЗ з єдиним центром обробки даних з'явилась нова автоматизована підсистема у складі АСК ВП УЗ - "Динамічна робота станційного вузла" (розробник ДП ПКТБ АСКЗТ

України). Вона дозволила усунути недоліки попередніх версій системи: послідовна обробка інформаційних потоків від АРМ-ів станції - спочатку в АСК СС, потім в АСК ВП УЗ, що погіршувало час реакції системи; відсутність даних по вагонах і даних з перевізних документів. До кінця 2011 року система впроваджена на всіх сортувальних станціях країни [34]. Етапи розвитку систем автоматизації сортувальних станцій представлені на рис. 1.5.

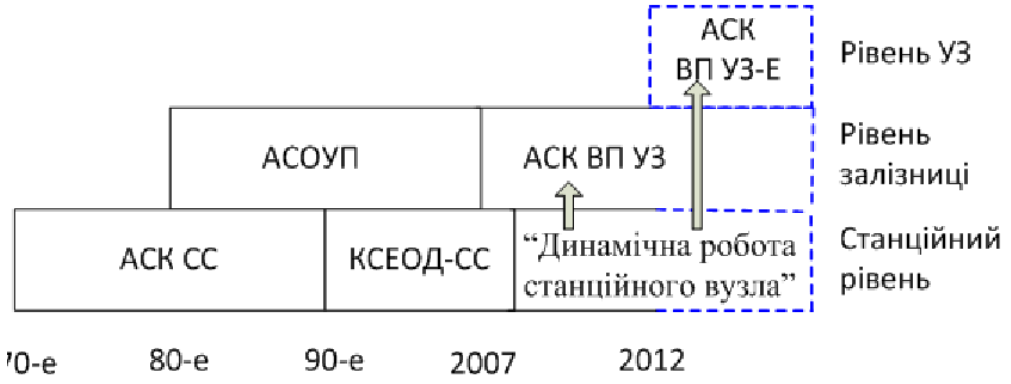


Рис. 1.5 Етапи розвитку систем автоматизації сортувальних станцій

Нові ресурсоощадні системи автоматизації сортувальних станцій мають інтегруватися в єдину систему на основі АСК ВП УЗ-Є. Тому необхідно дослідити, які ресурсні резерви має ця система.

1.5.1 Чотирирівнева структура АСК ВП УЗ-Є

Для дослідження характеристик АСК ВП УЗ-Є будемо використовувати повну 4-х рівневу модель системи, представлену на рис. 1.6.

Комплексування обчислювальних ресурсів на різних рівнях здійснюється або шляхом побудови багатомашинних систем (кластерів, як це показано на рис. 1.3, або шляхом використання багатопроцесорних систем (симетричне мультіпроцесування - SMP). Ефективність цих варіантів визначається кількістю процесорів, що використовуються. На

рис. 1.7 (за даними www.tpc.org) показана продуктивність систем на суміші (бенчмарці) TPC-C (у кількості транзакцій за хвилину - tpmC), що припадає на один процесор [204]. TPC-C припускає, що ≥ 90% транзакцій мають обмеження на час відповіді (в середньому 5 секунд).

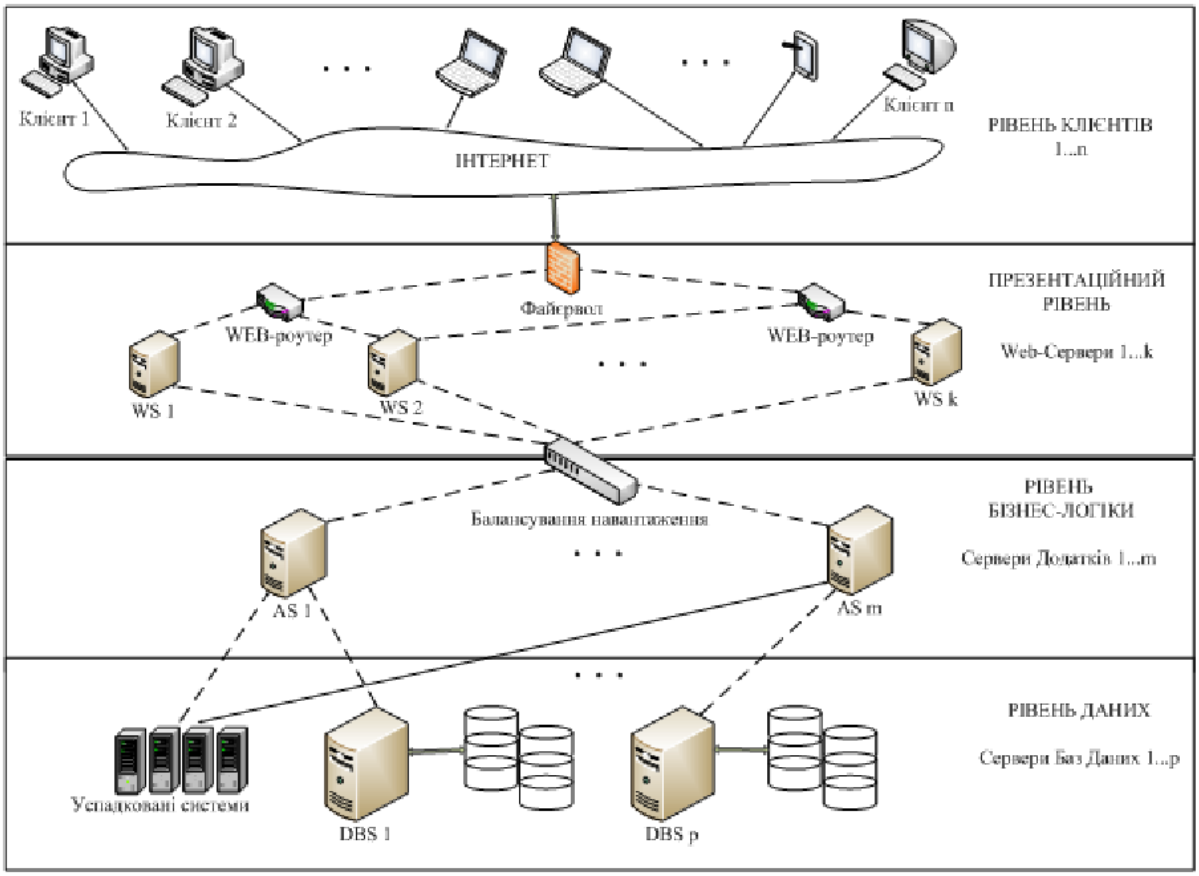


Рис. 1.6 Чотирирівнева структура АСК ВП УЗ-Є

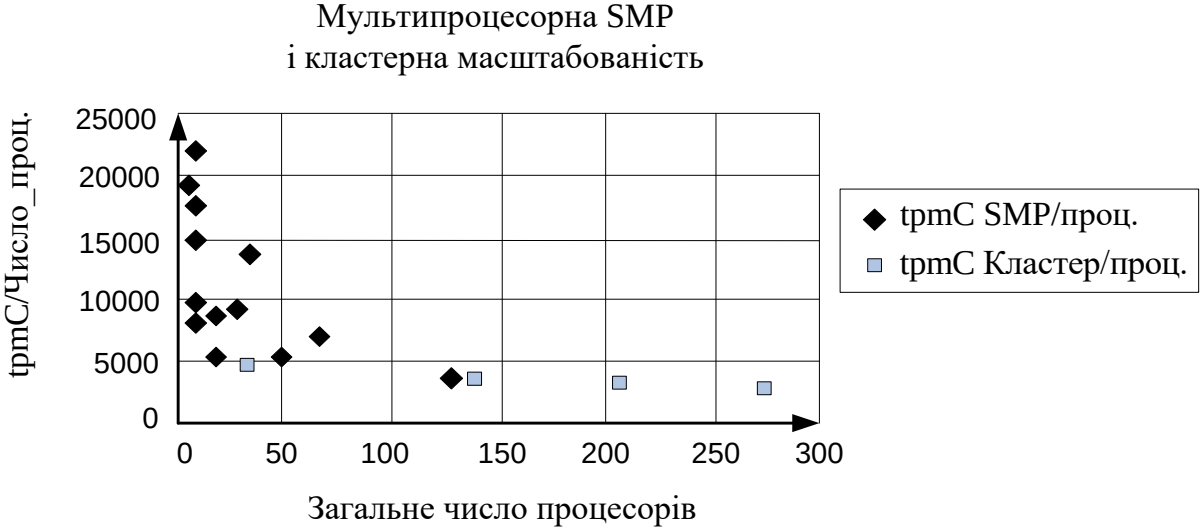


Рис. 1.7 SMP і кластерна масштабованість

Таким чином, для системи АСК ВП УЗ-Є можна рекомендувати при великій кількості серверів використовувати кластеризацію (наприклад, на рівнях WS, AS) і технологію SMP при кількості серверів - до 15 (рівень DBS).

У недовгій історії розвитку комп'ютерних систем можна чітко відокремити ряд основних етапів, які характеризуються переважною архітектурою побудови таких систем (див. рис. 1.3):

1 - HC-FS;

2 - HC-DBS;

3 - SC-AS-DBS;

4 - SWC-WS-DBS;

5 - архітектура з чотирма і більше рівнями (SWC-WS-AS-DBS).

Цей перелік архітектур можна доповнити ще двома:

6 - однорангова (пірінгова, p2p) архітектура;

7 - сервіс-орієнтована архітектура;

8 - мережецентрична сервіс-орієнтована архітектура.

Кожна нова архітектура з'являється внаслідок підвищення складності інформаційних систем: розмір збереженої інформації щороку подвоюється, число користувачів систем збільшується в геометричній прогресії, з'являються різні сервіси, які працюють на різних платформах і т.д. Сучасним вирішенням цієї проблеми є мережецентричний підхід до проектування інформаційних систем, в основі якого лежить сервіс-орієнтована архітектура.

Сервіс-орієнтована архітектура (service-oriented architecture, SOA) - модульний підхід до розробки інформаційних систем, в основі якого лежить розробка сервісів із стандартизованими інтерфейсами [43, 55].

Відзначимо лише одну основну характеристику мережецентричних систем, яка важлива для ІС реального часу. Це швидкість прийняття рішень (speed of command) - час, необхідний для проходження повного циклу Бойда "Спостереження - Орієнтація - Рішення - Дія" (Observe -

Orient - Decide - Act, OODA) [43]. Такий цикл включає в себе: збір інформації з внутрішніх і зовнішніх джерел - спостереження; формування множини можливих варіантів і оцінка кожного з них по сукупності критеріїв - орієнтація; вибір найкращого плану дій для практичної реалізації - рішення; практична реалізація обраного плану дій - дія. Очевидно, що кількість рівнів ієрархії або етапів у прийнятті рішення по OODA не повинно перевищувати чотири. Основу таких систем складають Центри обробки даних (ЦОД) [127, 141].

В даний час АСК ВП УЗ-Є можна представити як мережецентричну, сервіс-орієнтовану, 4-х рівневу інформаційну систему реального часу, модель якої показана на рис. 1.8. Її можна розглядати як перехідну до "хмарних обчислень" - динамічним IT-сервісам на базі WEB [51, 155].

Розглянемо тепер первинні дані для моделювання АСК ВП УЗ-Є.

1.5.2 Обчислювальні і комунікаційні ресурси системи

Система АСК ВП УЗ-Є реалізована на базі моделі P780 фірми ІБМ [16], яка має такі основні характеристики (максимальна конфігурація): вісім 3.86 GHz POWER7 восьмиядерних процесорних модулей; 64 ядра 3.86 GHz 256 KB L2 cache на ядро; 8 MB L3 cache на ядро (eDRAM); до 1 TB 1066 MHz DDR3 або до 2 TB 800 MHz DDR3; Active Memory Expansion; до 24 дисків SFF або до 24 дисків SFF SAS, чотири slimline для SATA DVD-RAM; 24 PCI Express x8; операційні системи AIX, SLES, RHEL.

У моделях максимальний обчислювальний ресурс системи становить 64 процесори і 24 звичайних SATA або SAS-дисків (Serial Attached SCSI). SAS-диски мають більш високі швидкості виконання дискових операцій введення-виведення.

Комунікаційні ресурси системи мають наступні характеристики.

Швидкості внутрішніх локальних мереж в межах верхніх рівнів AS-

DBS: 10/100 Мбіт / с, 1 Гбіт / с, 2 Гбіт / с (див. рис. 1.3 у додатку І).

Що до зовнішніх каналів, то на даний час на залізницях України використовуються 2-х рівневі волоконно-оптичні лінії зв'язку (ВОЛЗ): магістральний рівень (STM-16) с максимальною пропускною спроможністю до 2,5 Гбіт / с; дорожній рівень (STM-4) з максимальною пропускною здатністю до 622 Мбіт / с. Для системи АСК ВП УЗ виділялася пропускна здатність 10 Мбіт / с [52]. Але у вузлах МПД, де немає ВОЛЗ, використовуються виділені і комутовані канали тональної частоти, канали мобільного та супутникового зв'язку (GRPS, EDGE). Середня пропускна здатність цих каналів складає від 19, 2 до 56 Кбіт / с, чого явно недостатньо для роботи в АСК ВП УЗ-Є.

Зовнішні канали передачі даних в новій системі АСК ВП УЗ-Є - 50 Мбіт / с з дорогами, на рівні доріг канали закільцьовані [10]. Наведені вище дані говорять, що існують достатні резерви підвищення пропускної спроможності зовнішніх каналів. Тому надалі зупинимося тільки на потрібних обчислювальних ресурсах.

1.5.3 Особливості побудови та функцінування

Розробка основних проектних рішень по створенню єдиної АСК вантажними перевезеннями здійснювалася виходячи з двох основних положень [147]:

- збереження загальної методології (ідеології) АСК ВП УЗ-Є;
- побудова системи з єдиним централізованим серверним вузлом.

Одним з основних принципів побудови АСК ВП УЗ-Є є декомпозиція системи на множину технічних, програмних та інформаційних компонентів (див. визначення поняття архітектура в даній роботі - (1.1)) з максимальною стандартизацією взаємодії всіх компонентів, що забезпечує:

- максимальну відкритість системи до нарощування функціональних можливостей (поява нових завдань і модифікація існуючих);

- простоту супроводу і розвитку системи;

- можливість розпаралелення розробки.

Функціональні проектні рішення в системі згуртовано по функціональним комплексам (ФК): ФК організації перевізного процесу; ФК комерційної роботи; ФК вагонного господарства; ФК локомотивного господарства.

1.5.4 Граничні проектні оцінки характеристик системи

Всі проектні рішення розроблялися виходячи з таких оцінок розмірностей системи, які можна використати для оцінки "граничних" характеристик системи:

кількість залізничних абонентів, що беруть участь у формуванні БД - до 20 тисяч; вони забезпечують середньодобове оновлення бази даних - до 500 тисяч (навантаження на AS-DBS: 5,79 транзакцій/с; приймаємо $t_{\text{обр}}^{\text{AS}} = 0,2\,\text{с}$ $t_{\text{обр}}^{\text{DBS}} = 0,6\,\text{с}$);

кількість залізничних абонентів, які користуються інформаційно-довідковою системою - до 25 тисяч; середньодобова кількість звітних документів, що видаються - до 1 мільйона (навантаження на AS-DBS: 11,58 транзакцій/с; будемо припускати, що $t_{\text{обр}}^{\text{AS}} = 0,1\text{с}$ $t_{\text{обр}}^{\text{DBS}} = 0,3\text{с}$);

кількість взаємодіючих АСК клієнтів - до 10 тисяч; середньодобова кількість транзакцій - 60 тисяч (навантаження на AS: 0,69 транзакцій /с; приймаємо $t_{\text{обр}}^{\text{AS}} = 0,2\,\text{с}$).

Визначимо граничні характеристики системи АСК ВП УЗ-Є на основі введених оцінок. Часові характеристики обробки транзакцій, наведені вище, прийняті за даними [10, 124, 127, 141]. Розрахункова кількість процесорів в кластерах за рівнями - $K^{\text{WS}} = 1;\ K^{\text{AS}} = 6;\ K^{\text{DBS}} = 12.$

В якості початкового рівня вхідних потоків приймемо ті, які були використані в проекті (на рис. 1.8-1.111 це відповідає значенню 1). При

цьому 6 серверів в кластері AS забезпечують 50% запас по середньому завантаженню (див. рис.1.8), а ось в кластері DBS 12 серверів забезпечують всього 5% запас до рівня (див. рис. 1.9).

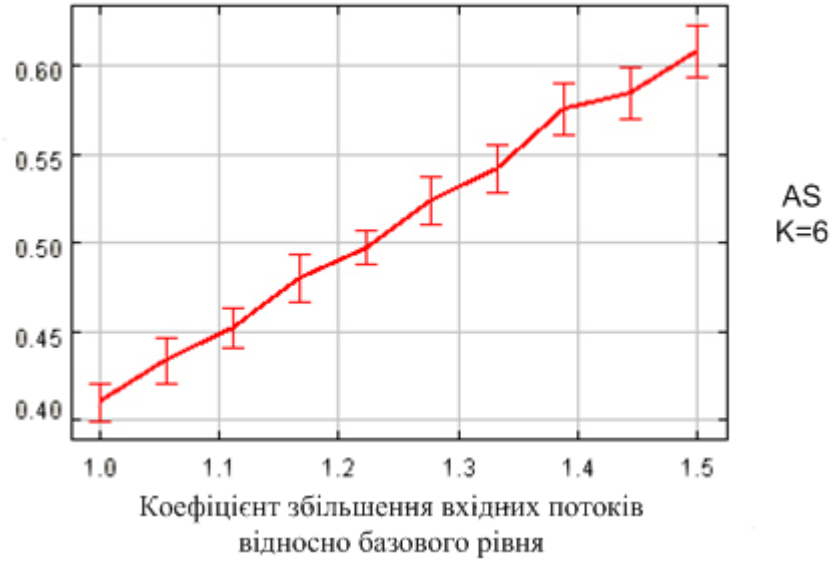


Рис. 1.8 Середнє завантаження кластера AS

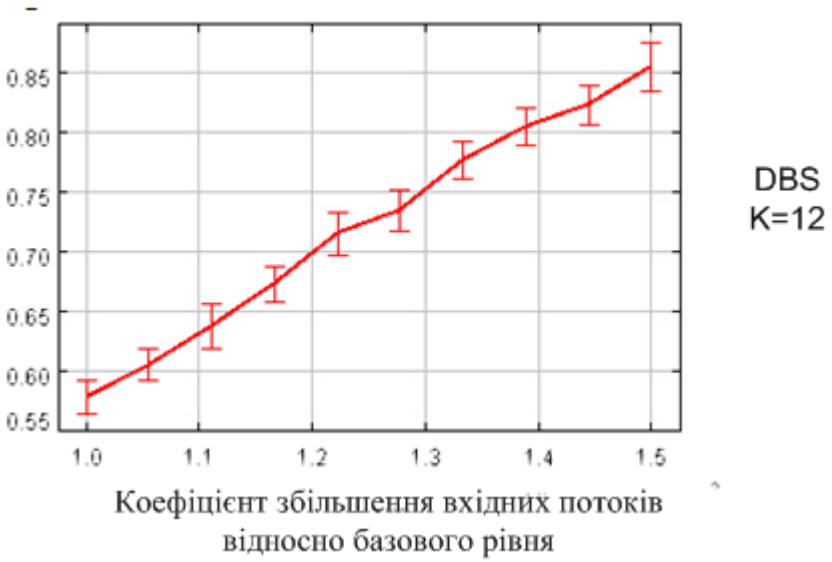


Рис. 1.9 Середнє завантаження кластера DBS

Оскільки АСК ВП УЗ-Є розглядається як система реального масштабу часу, то досить важливим параметром може бути час відгуку системи. На рис. 1.10 і 1.11 показана залежність часу реакції від вхідних потоків. Якщо для серверів додатків вона має практично лінійний вигляд, і є достатній запас в прийнятному діапазоні часу (див. рис. 1.10), то для серверів баз

даних крива має експоненційний вигляд і при 50% навантаженні (відносно базового рівня) досягає критичних значень (див. рис. 1.11). Відзначимо, що це в тому випадку, якщо будуть збільшені проектні граничні параметри розмірності системи [147]. Таким чином, потрібні "проектні" обчислювальні ресурси (19 процесорів) з надлишком перекриваються ресурсами системи P780 (64 процесори).

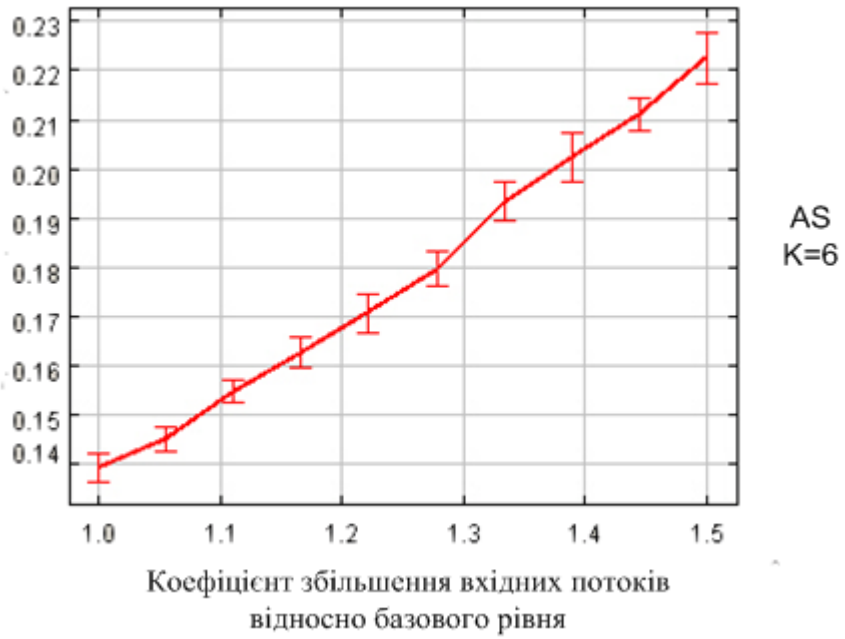


Рис. 1.10 Середній час відповіді кластера AS

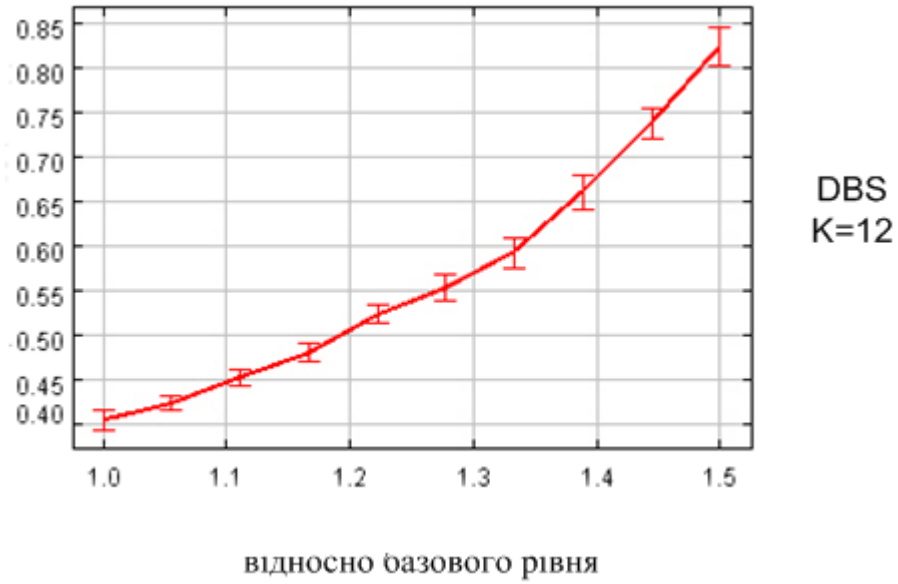


Рис. 1.11 Середній час відповіді кластера DBS

1.5.5 Оцінка експлуатаційних показників АСК ВП УЗ-Є

Система АСК ВП УЗ-Є введена в промислову експлуатацію 7 липня 2012 року і вже відомі перші результати її експлуатації, які можна використовувати для моделювання і дослідження системи [10, 124, 127, 141]. При цьому вхідні потоки дещо відрізняються від їх проектних оцінок [147].

В даний час з системою працюють 25 тисяч користувачів з 12 господарств залізниць України, які формують вхідний потік запитів від 650 тисяч до 1 млн. на добу, що відповідає інтенсивності 7,5 - 11,6 транзакцій/с. Потоки в системі розподіляються наступним чином: на перших двох рівнях WS/AS - 0,3/0,7; далі, на AS/DBS - 0,45/0,55.

Середній час обробки заявок на процесорах AS-DBS розподілимо пропорційно складності операцій у відповідних серверах $t_{\text{обр}} = t_{\text{AS}} + t_{\text{DBS}} = 0,2 + 0,6 = 0,8\text{с}$.

Середній розмір повідомлень, що надходять в систему, приймемо 21 Кбайт, розподіл розмірів повідомлень - за законом Парето [204].

У даній роботі розглянемо макромодель системи, яка дозволяє визначити потрібні обчислювальні ресурси за рівнями WS-AS-DBS з урахуванням розвитку АСК ВП УЗ-Є (збільшення вхідного потоку заявок, ускладнення і збільшення кількості додатків в системі). Виконані дослідження на аналітичних моделях M/M/K і Par/M/K [101] дали наступні наближені результати (без структуризації запитів, без оцінки складності додатків і врахування особливостей роботи з дисковими масивами.

При моделюванні передбачалися наступні базові параметри обробки запитів: $t_{\text{обр}}^{\text{WS}} = 10$ мс; $t_{\text{обр}}^{\text{AS}} = 200$ мс; $t_{\text{обр}}^{\text{DBS}} = 600$ мс. При цьому кількість процесорів в кластерах за рівнями: $K^{\text{WS}} = 1$; $K^{\text{AS}} = 5$; $K^{\text{DBS}} = 5$. На графіках (рис.1.12) показані резерви по ускладненню задач обробки запитів до рівня, коли завантаження досягає 0,6.

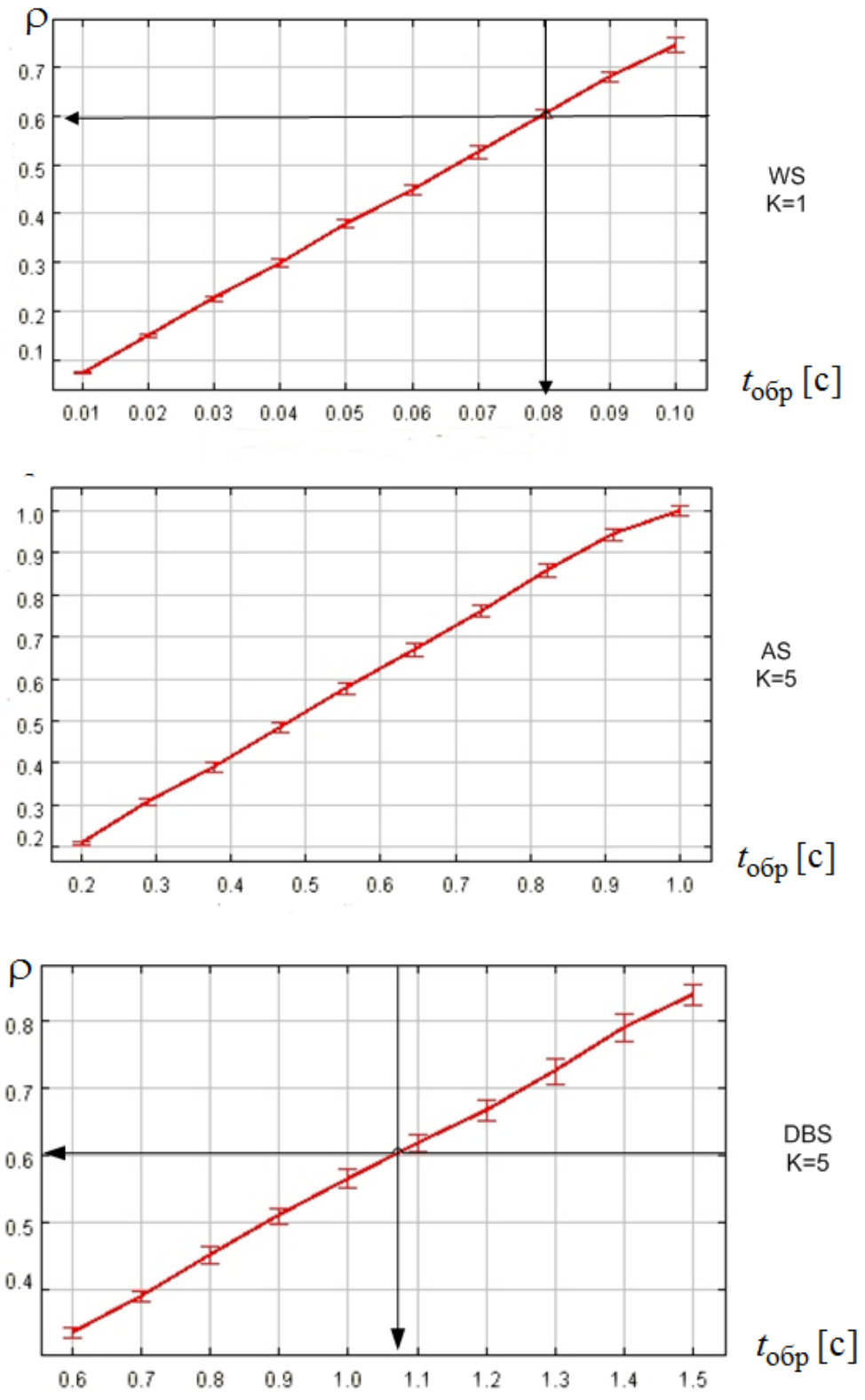


Рис. 1.12 Середні завантаження процесорів по кластерам системи в залежності від часу обробки запитів

У цьому випадку слід розширювати кластери, для чого є всі необхідні резерви в системі P780.

Застосування розподілу Парето для зовнішніх заявок не впливає на характеристики рівнів AS-DBS, так як згладжується на рівні WS. Тому в подальших дослідженнях можна користуватися експоненціальним розподілом і моделями М/М/К.

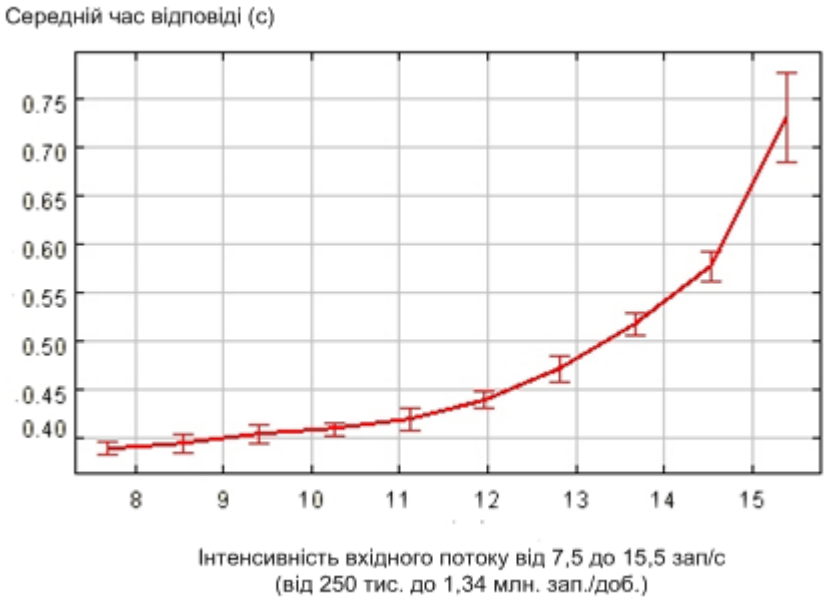


Рис. 1.13 Залежність середнього часу відповіді системи від вхідного потоку запитів

Для системи реального часу середній час відповіді може бути однією з головних характеристик, для якої встановлюються певні вимоги замовника. У цьому випадку можна скористатися залежностями, вид яких відображено на рис.1.13.

Отримані результати показують, що система АСК ВП УЗ-Є в процесі перших місяців експлуатації використовує тільки 60% від проектних ресурсів і 17% від обчислювальних ресурсів системи Р780. Таким чином, є всі необхідні передумови для її розширення завданнями планування і керування процесами розформування-формування поїздів на сортувальних станціях.

РОЗДІЛ 2
ТЕОРЕТИЧНІ ОСНОВИ ДЕКОМПОЗИЦІЇ СИСТЕМ АВТОМАТИЗАЦІЇ ТЕХНОЛОГІЧНИХ ПРОЦЕСІВ

2.1 Особливості проектування складно структурованих систем керування

Процес проектування пристроїв автоматизації завжди починається з аналізу системи керування, для якої вони створюються. Дослідженню підлягають такі її характеристики як кількість контрольованих параметрів, час реакції керованого об'єкту на керуючу дію, необхідна точність керування. Окрім них важливе значення має математична модель функціонування системи - сукупність співвідношень, що визначають залежності зміни регульованих величин від вхідних дій, що формуються обчислювальним комплексом.

Функціонування системи можна представити вектор-функцією $\mathbf{S}(t)$, значення якої в моменти часу $t_0,...,t_i,t_{i+1},...$ визначають стани системи $\mathbf{S}(t_0),...,\mathbf{S}(t_i)$, $\mathbf{S}(t_{i+1}),...$. Кожен стан характеризується множиною величин, що описують зовнішні чинники (вони називаються діями), що впливають на систему, і протікання процесів усередині самої системи.

Призначення системи керування - цілеспрямована зміна стану об'єкту управління (рухомих відчепів), або, що те ж саме, управління процесом функціонування об'єкту - сортувальної гірки. Щоб керувати, необхідно знати: як поводяться об'єкти управління (стани відчепів в задані моменти часу); які ті зовнішні дії на відчепи, якими ми не можемо керувати (дії зовнішнього середовища); яка мета керування; які засоби дії на об'єкт ми маємо (які ресурси є в нашому розпорядженні - сповільнювачі, стрілочні переводи тощо).

На рис. 2.1 показані взаємозв'язки об'єкту керування - сортувальної гірки, системи керування і навколишнього середовища. Дії, що

виробляються системою керування будемо називати *керуючими діями.* На рис. 2.1 вони позначені $\mathbf{X} = (x_1, x_2, \ldots, x_k)$. Дії, не залежні, від керуючої системи, будемо називати *обуреннями.* Частина їх вимірюється (контрольовані обурення) $\mathbf{Z} = (z_1, z_2, \ldots, z_o)$, частина з різних причин не вимірюється (неконтрольовані обурення) $\mathbf{F} = (f_1, f_2, \ldots, f_n)$, причому точне число неконтрольованих обурень зазвичай невідоме. Контрольовані величини, по яких ведеться керування, називатимемо *керованими* або *регульованими*. Вони позначені $\mathbf{Y} = (y_1, y_2, \ldots, y_r)$.

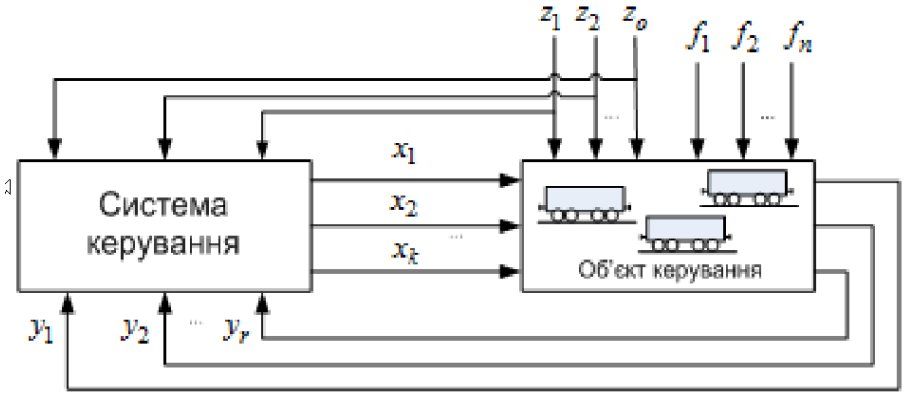


Рис. 2.1 Взаємозв'язки об'єкта керування, системи керування і навколишнього середовища

У системах керування сортувальними гірками реалізуються, як правило, найбільш складні завдання адаптивного керування, у яких керуючі дії, або алгоритм керування, автоматично і цілеспрямовано змінюється для здійснення в якому-небудь сенсі якнайкращого управління об'єктом, причому характеристики об'єкту або дії зовнішнього середовища можуть змінюватися наперед непередбаченим чином.

У всіх випадках керуюча система визначає необхідні керуючі дії відповідно до певного закону керування $\mathbf{X}(t) = \Phi(\mathbf{Z}, \mathbf{Y}, t)$. Для реалізації складних законів керування потрібні складні алгоритми і програмні комплекси, відмінністю яких є наявність жорстких обмежень на час рішення задач керування, що обумовлене високою швидкістю зміни обурень $\mathbf{F}(t)$, що діють на об'єкт керування.

Практика показує, що через згадані вище особливості, а також із-за наявності жорстких вимог і обмежень, проектування інформаційно-керуючих комплексів для автоматизації сортувальних гірок є найбільш складним інженерним завданням.

Відмінності в характері і складності керованих об'єктів (відчепів) не можуть не позначатися на вимогах до характеристик і структури АСК, що, у свою чергу, впливає на вибір параметрів і методику проектування обчислювальних комплексів, що входять до її складу. Для загальної оцінки цього впливу виділимо найважливіші показники і розглянемо їх особливості.

Кількість контрольованих параметрів. До них відносяться вхідні дії $\mathbf{X}(t)$ і $\mathbf{Z}(t)$, регульовані величини $\mathbf{Y}(t)$, а також деякі параметри, що описують стан об'єкту управління, причому частина їх не змінюється в процесі управління або змінюється дуже рідко (практично постійні параметри). Як було визначено в першому розділі, $\mathbf{Z}(t)+\mathbf{Y}(t)$ є умовною інформаційною потужністю (УІП) об'єкта керування. За діючими стандартами сортувальні гірки залежно від об'єкту керування (кількості пучків на гірці) відносяться до систем підвищеної і великої (максимальної) УІП з кількістю вхідних сигналів від 809 до 2153.

Із зростанням складності об'єкту керування або умов зовнішнього середовища, в яких функціонує об'єкт, кількість контрольованих параметрів збільшується. Із зростанням їх кількості зростає складність прийняття рішення по керуванню, оскільки збільшується число чинників, що враховуються, і різко підвищується кількість взаємозв'язків між ними і регульованими величинами. Це приводить до зростання кількості і складності алгоритмів керування; ускладнюється діяльність оператора з аналізу обставин і ухвалення рішення; підвищуються вимоги до швидкості технічних засобів, які збирають і обробляють інформацію про контрольовані параметри, формують сигнали керування; підвищуються вимоги до місткості запам'ятовуючих пристроїв для зберігання необхідних в процесі керування величин.

Для обробки інформації створюються складні алгоритми і комплекси програм, причому їх склад в процесі експлуатації практично не змінюється і послідовність рішення окремих задач жорстко задана. Тому, надалі будемо говорити про φ-транзакції як послідовність завдань формування $x(t_i) \in \mathbf{X}(t)$ (див. розділ 4).

Залежність регульованих величин від вхідних дій. Характер і математичне формулювання цієї залежності визначають складність алгоритмів і програм керування. В АСК сортувальної гірки зв'язок між змінними, що описують керований рух відчепів, задається диференційними рівняннями, з параметрами, залежними від вхідних і вихідних величин.

Динамічні характеристики і часові обмеження. Вони визначаються швидкістю зміни стану об'єкту керування (руху відчепів) $\mathbf{S}_{\text{вчп}}(t)$ і залежать від його інерційності і швидкості зміни обурень.

Хай t_i і t_{i+1} - два моменти часу, в які замірені обурення $\mathbf{Z}(t_i)$ і $\mathbf{Z}(t_{i+1})$, що вимагають видачі відповідних керуючих дій. За час $t_{i+1} - t_i$ повинна бути оброблена необхідна інформація, прийняте рішення по керуванню, видані керуючі сигнали, початий і закінчений перехідний процес зміни стану від $\mathbf{S}_{\text{вчп}}(t_i)$ до обчисленого в процесі прийняття рішення стану $\mathbf{S}_{\text{вчп}}(t)$, де $t_i < t \le t_{i+1}$. Залежно від постановки завдання оптимального керування довжина інтервалу $t - t_i$ часто повинна задовольняти певним вимогам (наприклад, може бути поставлене завдання її мінімізації).

У цій послідовності дій виділяються дві складові: прийняття рішення керуючою системою (час $t_{\text{пр}}$), і відпрацювання прийнятого рішення об'єктом керування (час його реакції $t_{\text{р}}$). Очевидно, що $t_{\text{пр}} + t_{\text{р}} \le t - t_i$. Якщо об'єкт керування багато зв'язний (керування декількома відчепами), то вектори **X**, **Y** і **Z** мають декілька координат, тобто це співвідношення повинне виконуватися для кожної координати. Із зростанням швидкості

зміни обурень довжина інтервалу $t - t_i = \Delta t$ зменшується, що при незмінних динамічних характеристиках об'єкту приводить до зменшення допустимого часу $t_{\text{пр}}$, відведеного для прийняття рішення, оскільки

$$0 < t_{\text{пр}} \le \Delta t - t_{\text{р}}. \qquad (2.1)$$

Це, у свою чергу, викликає підвищення вимог до швидкодії технічних засобів збору, обробки і видачі інформації і вимог до надійності функціонування системи, оскільки витрати часу на відновлення можуть привести до порушення умови (2.1) і зміни ступеню автоматизованого керування (у автоматизованих системах).

У АСК сортувальною гіркою вагонні сповільнювачі верхньої і групової позицій повинні мати час відпускання не більше 0,8 с, а час переведення стрілки, включаючи час спрацьовування комутуючої апаратури, повинен бути не більше 0,8 с [14]. Для окремих пристроїв ці значення можуть бути жорсткішими. Так, наприклад, для сповільнювачів РНЗ-2М час спрацьовування при розгальмовуванні (від подачі команди до повного зняття зусилля) не більше 0,6 с, а при переході від розгальмованого положення в загальмоване - не більше 0,7 с [6].

З (2.1) видно, що час, який відводиться керуючій системі для виконання розрахунків за програмами, що реалізовують алгоритм керування $t_{\text{ра}}$, повинен задовольняти умові:

$$t_{\text{ра}} \le t_{\text{пр}} < \Delta t - t_{\text{р}}.$$

Якщо обурення змінюються швидко (Δt мале), а завдання керування складні ($t_{\text{ра}}$ велике), то величина $t_{\text{ра}}$ відрізняється від $t_{\text{пр}}$ на малу величину ε, яка значно менше $t_{\text{ра}}$, причому час реалізації алгоритму порівняний з часом перехідного процесу об'єкту управління:

$$t_{\text{ра}} = t_{\text{пр}} - \varepsilon,\ \varepsilon \ll t_{\text{ра}},\ t_{\text{ра}} \approx t_{\text{р}}. \qquad (2.2)$$

В більшості випадків практична реалізація рівності в співвідношеннях (2.2) зв'язана з високими вимогами до швидкодії технічних засобів автоматизації.

Величина Δt може визначатися швидкістю зміни не тільки обурень, але і інших параметрів, що характеризують стан об'єкту (наприклад, просторових координат), а також заданими моментами реального астрономічного часу.

Із співвідношень (2.2) видно, що основним параметром, що визначає швидкості всіх процесів в системі керування, є величина $t_{\text{р}}$ - час перехідного процесу в об'єкті керування. Вона визначає, який масштаб часу слід вибрати при аналізі і синтезі системи.

Співвідношення (2.2) показує, що результати використовуються для управління практично негайно (через малий інтервал ε). Керуюча система в цьому випадку працює як би в тому ж масштабі часу, що і об'єкт керування, тобто в реальному часі.

Поняття масштабу реального часу, що характеризує особливості функціонування керуючої системи, часто поширюють і на систему керування в цілому. Говорять, що АСК працює в масштабі реального часу (або просто «АСК реального часу»), якщо хочуть підкреслити її наступні особливості:

1) надзвичайно малий час, відведений для ухвалення рішення, сумісний з часом перехідного процесу в об'єкті;

2) складність алгоритмів рішення задач управління і практично негайне використання результатів рішення для керування;

3) неприпустимість як передчасної видачі керуючих сигналів, так і їх запізнювання.

Точність управління. При проектуванні АСК цей показник оцінюється як ступінь відхилення результатів вимірювань або величин керуючих дій від деяких значень, визнаних ідеальними при певних припущеннях (ідеальний алгоритм обчислень, абсолютна точність

представлення початкових даних і результатів і т.п.) в даний момент часу. Точність управління також характеризується ступенем відхилення фактичних моментів видачі керуючих дій від оптимальних або заданих.

З підвищенням необхідної точності ускладнюються алгоритми рішення задач керування (наприклад, із-за збільшення ступеня полінома, що апроксимує характеристику датчика); збільшується кількість розрядів, що представляють інформацію, яка передається або обробляється; підвищуються вимоги до швидкодії обчислювальних комплексів і до швидкості передачі інформації.

Точність, як параметр, що впливає на вибір технічних рішень, повинні враховуватися і при проектуванні АСК технологічними процесами на сортувальних станціях. Порушення необхідної точності може привести до аварії об'єкту керування і тому розцінюється як відмова системи.

Розподіленість системи в просторі і умови протікання керованого процесу. Керований процес протікає в просторі і в часі. Зміна координат точок прийому і видачі інформації може бути пов'язана як з переміщенням самого об'єкту управління, так і з послідовним виконанням ланцюжка технологічних операцій на різному устаткуванні. АСК сортувальних станцій є розподіленими системами, тому важливе значення має система передачі даних.

Необхідність підходу до аналізу і проектування інформаційно-керуючих систем на сортувальних станціях як до систем реального масштабу часу вказувалась в роботах [105, 49, 44, 131], де вирішувались окремі задачі, пов'язані алгоритмізацією процесів керування і розробкою локальних підсистем керування.

Узагальнюючи всі розглянуті положення і аналізуючи їх з погляду вимог до обчислювальних комплексів, можна зробити наступні висновки.

1. Всі технічні засоби автоматизації, що працюють в складі АСК, орієнтовані на реалізацію певних алгоритмів, які потребують ретельного аналізу і оцінювання часу їх виконання.

2. АСК сортувальних станцій відносяться до класу систем реального масштабу часу і характеризуються:

жорсткими вимогами до часу формування рішень, величина якого близька до часу реалізації алгоритму, що робить необхідним не тільки оцінювати час виконання алгоритмів, але й інші втрати часу в чергах на обслуговування з пріоритетами в каналах обробки і передачі даних;

складними програмними комплексами для вирішення певних, наперед заданих завдань і великим об'ємом даних, що обробляються, які необхідно враховувати при проектуванні АСК;

зв'язком з автоматичними джерелами і приймачами інформації, що в розподілених системах актуальним робить завдання розробки раціональної інформаційної структури системи;

високими вимогами до надійності роботи, до достовірності інформації, що треба враховувати при розробці технічних структур і програмно-діагностичного забезпечення АСК;

обмеженнями на споживану енергію і габарити, що визначається при виконанні функціонально-логічного і конструктивного компонування уніфікованих технічних засобів автоматизації.

Перелічені вимоги слід враховувати в методиці проектування і вдосконалення технічних засобів автоматизації сортувальних станцій. І одним з головних завдань є пошук моделей, які дозволяють аналізувати дискретні системи реального масштабу часу керування сортувальними гірками і формувати принципи декомпозиції-композиції системи керування в умовах часових обмежень (2.2). В якості таких моделей в роботі пропонуються багатосортні алгебраїчні моделі дискретних систем реального часу.

2.2 Розробка багатосортних алгебраїчних моделей дискретних систем керування реального часу

Поняття «система реального часу» (real-time system) описує цифрову систему, в якій дискретний обчислювальний процес повинен реагувати на внутрішні і зовнішні події, що відбуваються у випадкові моменти часу [146, 42, 163]. Опис таких процесів включає час як незалежну змінну, що приймається практично у всіх теоретичних підходах. У системах реального масштабу часу (*СРМЧ*) змінна часу набуває дискретних значень і представляється різними способами, наприклад лічильниками (таймерами), що не завжди відповідає умовам безперервного реального часу. До того ж проблема ускладнюється в розподілених інформаційних системах (на сортувальних станціях), де виникає необхідність синхронізації або встановлення відповідності систем відліку часу. Характерною особливістю *СРМЧ* є встановлення обмеження на тривалість системного часу на обробку певної події, що виникла в підсистемі *i*, і пов'язаної з виконанням певної дії (наприклад, дій з управління записом в зовнішню пам'ять, з видачі сигналу керування, тощо) у віддаленій підсистемі *j*.

У даному підрозділі пропонується абстрактна модель для систем реального масштабу часу, яка поєднує в собі дискретний характер розподіленого обчислювального процесу з безперервним характером часу. *СРМЧ* вивчаються вже тривалий час в області систем керування і побудови АСКТП. Моделі і методи вирішення завдань для таких систем, запропоновані різними дослідниками, засновані на диференційних рівняннях, які є основним математичним апаратом в теорії управління з моменту її створення [180]. Зростання використання дискретних обчислювальних процесів привело до появи нових методів дослідження цього класу дуже важливих систем. Інтенсивно розвиваються часова логіка [202], алгебричні системи (I/O автомати) [184, 208, 205], алгебра процесів реального часу [170, 188].

У цій роботі пропонуються абстрактні моделі дискретних процесів реального часу, які можна використовувати при розробці принципів декомпозиції/композиції технологічних процесів на сортувальних станціях і відповідних інформаційних систем управління, для розробки інформаційно-часових діаграм загального алгоритму функціонування розподілених систем управління на станціях.

2.2.1 Час як змінна

Розглянемо більш детальніше загальні властивості *СРМЧ*. Точками консенсусу у всіх наукових роботах прийнято:

- всі моделі повинні поважати принцип причинності;

- система повинна реагувати на події або стимул-реакції;

- система має дискретне подання і реалізацію (у вигляді цифрових пристроїв).

Для цих властивостей базовим є розуміння сутності часу.

Поняття час може бути формалізоване декількома способами залежно від того, що для нас є важливим в дослідженні. У класичній теорії автоматів воно моделюється множиною натуральних чисел N, що виходить з визначення поняття автомат. Використовуючи таке подання, можна оцінити час роботи складної програми, виражаючи його в кількості переходів (див. розділ 4). Час може бути заданий явно і описуватися як спеціальна подія, що нескінченно повторюється, під назвою *clock* (годинник) [202], [172], або може бути явно представлено у визначенні поняття система як інший компонент системи [191]. Вважається, що час - природне розширення класичної теорії автоматів. Недолік такого визначення витікає з необхідності завдання спеціальних правил, які мають бути введені в цілях забезпечення розуміння часових подій. У даній роботі, як і у [188], вважатимемо, що час представляється множиною позитивних чисел R^+, і разом із звичайним бінарним відношенням $\leq$ описують модель

часу. Легко бачити, що R^+ представляє повну частково впорядковану структуру.

Час є відносним по відношенню до деякої системи відліку, в якій він вимірюється, і, таким чином, кожна система має свій локальний (місцевий) час *local time (LT)*. Для синхронізації роботи різних підсистем в розподіленій системі вводиться поняття системного (глобального) часу - *global time (GT)*. Між будь-якими *LT* і *GT* існує ньютонівське часове відношення. Якщо τ є момент часу в *GT*, коли система *S* почала працювати, то цей момент часу в зв'язаних підсистемах відповідатиме t_g в *GT* і t_l в LT, які пов'язані виразом $t_g = t_l + \tau$.

Розглянемо систему *S*, що складається з 2-х підсистем P_1 і P_2. Кожна підсистема має свій власний локальний час T_1 і T_2 відповідно. Передбачимо, що обоє підсистеми починають виконувати свої функції в деякий момент τ глобального часу *GT*. Зазвичай, між T_1 і T_2 існує ізоморфізм: $id : T_1 \to T_2$, який показує, що час в підсистемах вимірюється в тому ж порядку.

Системи з дискретними подіями [198] характеризуються тим, що події з'являються в дискретні моменти часу і ці моменти можуть бути в будь-якому місці часової осі. R^+ є безперервною послідовністю чисел від нуля до ∞ і тому кожній події відповідатиме своя точка (момент) на осі часу.

Часові структури представляють не лише точки (моменти) часу на часовій осі. Важливою характеристикою є часові інтервали. Позначатимемо часовий інтервал $[t_1, t_2)$ як $\sigma_{t_1}^{t_2}$. Залежності між t_1 і t_2 дозволяють виділити наступні спеціальні випадки, для яких введемо відповідні позначення.

Часовий інтервал, який має обмеження зверху $[0, t)$ позначатимемо σ^t. Аналогічно, інтервал, який має лише нижнє обмеження $[t, \infty)$,

позначатимемо σ_t. Одноточкові інтервали вигляду $[t,t]$ позначатимемо $\overline{\sigma_t}$.

Розглянемо тепер множину Σ всіх часових інтервалів на R^+. На цій множині визначимо часткову операцію «+», яку назвемо об'єднання інтервалів (interval concatenation), що виконується таким чином (див. рис. 2.2):

$$\sigma_{t_1}^{t_2} + \sigma_{t_2}^{t_3} = \sigma_{t_1}^{t_3}, \tag{2.3}$$

$$\sigma_{t_1}^{t_2} + \sigma_{t_3}^{t_4} = \begin{cases} \sigma_{\min(t_1,t_3)}^{\max(t_2,t_4)}, \text{ якщо } [t_1,t_2) \cap [t_3,t_4) \neq \varnothing \\ \text{не визначено, якщо } [t_1,t_2) \cap [t_3,t_4) = \varnothing \end{cases}. \tag{2.4}$$

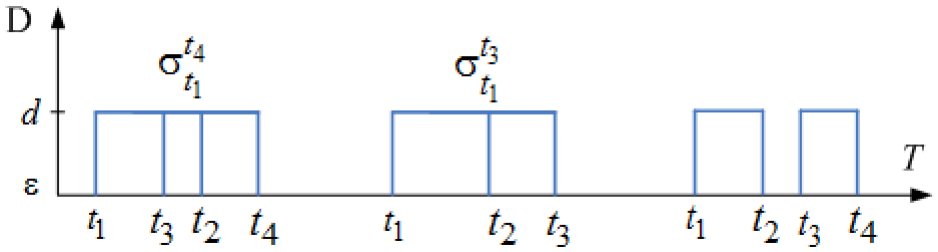


Рис.2.2 Об'єднання інтервалів, часові події

2.2.2 Події і принципи декомпозиції дискретних систем керування сортувальними гірками

Ґрунтуючись на часі існування подій, їх класифікуватимемо на *дискретні (discrete)* і *тривалі (duration)* події (наприклад, управління стріловим приводом і управління гальмуванням відчепів в сповільнювачі). У даній роботі дискретна подія в моделях буде базовим типом, а тривалу подію представимо як послідовність (рядок) дискретних подій. Введемо позначення: *dt* - дискретні події; *it* - інтервальні події.

Події визначатимемо двома часовими константами t_1 і t_2, які задають нижню і верхню межі і визначають валідність (область існування, час життя) подій (див. рис. 2.2).

Подію можна описати трійкою (e,t_1,t_2). Перша компонента $e \in D$, де D - домен подій, $t_1,t_2 \in R^+$ - часові величини, що визначають час t_1, коли подія стартувала, почала впливати на систему до тих пір, поки не настає момент часу t_2 завершення події. Події описуються у власній системі відліку. Аби бути точнішим, то якщо подія (e,t_1,t_2) з'являється в момент t_d за локальним часом системи, то вона має бути оброблена за інтервал часу $[t_d + t_1, t_d + t_2)$, аби отримати позитивний ефект (наприклад, встигнути перевести стрілку за маршрутом скачування відчепа на гірці). Цей же інтервал в *GT* має вигляд $[\tau + t_d + t_1, \tau + t_d + t_2)$.

Визначення 2.1 Множина часових інтервалів в домені D - це множина триплетів (e,t_1,t_2), де $t_1,t_2 \in R^+$, що володіє властивістю $t_1 \le t_2$, а саме $E = \{(e,t_1,t_2) \mid e \in D, t_1 \in R^+, t_2 \in R^+, t_1 \le t_2\}$.

Динаміка системи визначається взаємодією з довкіллям через свої входи і виходи. Хоча в реальних системах множина подій *E* складається з трьох підмножин (піддоменів): E^w - власне множина вхідних подій із зовнішнього середовища (зовнішнього світу, world); E^{sys} - множина внутрішніх, системних подій (наприклад, сигнали переривань по параметрах, що характеризують стан системи); E^{timer} - множина подій, що ініціюються за станом таймера (контрольно-діагностичні процедури, оновлення баз даних та ін.). Вочевидь, що $E = E^w \bigcup E^{sys} \bigcup E^{timer}$. Особливу увагу необхідно приділяти E^w, подіям, що формуються в зовнішньому середовищі. Ця множина є динамічною і такою, що складно структурується, для технологічних процесів на сортувальних станціях і, особливо, на сортувальних гірках. У останньому випадку $E^w_{\text{гірка}} \subset E^w$ може бути представлено у вигляді динамічного списку подій, що формується *i*-м відчепом, що скачується по гірці за *m*-м маршрутом. Кількість списків в технологічному циклі буде не менше числа відчепів в потязі, що

розформовується. У кожному списку будуть три види (підмножини) подій, що динамічно змінюються:

- $(e_{im}, t_1, t_2, \text{mod})$ прогнозовані події для відчепа i, що скачується за маршрутом m;

- $(e_{im}, t_1, t_2, \text{now})$ поточні події для того ж відчепа;

- $(e_{im}, t_1, t_2, \text{past})$ події, що мали місце з відчепом i, що скачується за маршрутом m, до моменту часу t_1.

Списки *mod* формуються перед розпуском чергового составу в результаті моделювання з метою досягнення мінімального часу розпуску. Списки past фіксують результати розпуску з метою корегування програми моделювання.

Динамічний характер цих списків обумовлений нештатними ситуаціями на гірці, пов'язаними, наприклад, з нагонами, не переведенням стрілок і створенням чужаків, і тому подібне. У цих випадках події *mod* і *now* повинні коригуватися в реальному часі, що вимагає відповідних обчислювальних ресурсів для не порушення граничного часу t_2. Розглянутий підхід до побудови списків подій пов'язаний з принципами декомпозиції/композиції розподіленої дискретної системи керування реального часу і принципами управління технологічним процесом розпуску составі на гірці [96].

Об'єктно-топологічний принцип декомпозиції, коли кожен список формується для об'єкту керування, яким є відчеп, що скачується за певним маршрутом відповідно до топології сортувальної гірки. Така організація подій передбачає наявність свого локального часу для кожного об'єкту (відчепа) зі своєю точкою відліку, тобто відчеп «живе» при скачуванні в своїй системі часових координат (LTO_i), але в рамках *GT*.

Другий спосіб організації $E^w_{\text{гірка}} \subset E^w$ відповідає принципу, який назвемо **принципом технологічної декомпозиції**, коли події пов'язані з технологічними ділянками певного типу, тобто функціонального

призначення з точки зору керування процесом розформування потягів на гірці. Наприклад, стрілочні ділянки, ділянки гальмівних позицій, контролю розчепа та ін. На кожній ділянці, для реалізації певної функції f_j, встановлений датчик j (один або декілька), що формує відповідну подію (e^{f_j}, t_1, t_2).

Третій принцип декомпозиції можна розглядати як комбінацію перших двох принципів - **принцип функціонально-топологічної декомпозиції** (ділянки групуються за топологією гірки). При цьому зберігається схема формування подій другого принципу декомпозиції - (e^{f_j}, t_1, t_2).

Четвертий принцип декомпозиції - **подійний** - полягає в формуванні класів подій (e^s, t_1, t_2), що потребують однотипної обробки на апаратно-програмно-алгоритмічному рівні. Тоді клас подій s містить k подій $E^{e_s}_{\text{гірка}} = \{e_{s1}, e_{s2}, ..., e_{sk}\}$ і для всіх класів S маємо $E^{w}_{\text{гірка}} = \{E^{e_s}_{\text{гірка}} \mid \forall s \in S\}$.

Вочевидь, що способи декомпозиції/композиції технологічного процесу і керуючої *ДСРЧ* взаємозв'язані між собою як з точки зору часових змінних, так і списків подій, що формуються.

Коли система отримує подію, цей момент стає частиною цієї події. Як приклад, розглянемо множину E у вище даному визначенні. Множина можливих рядків подій, які можуть бути генеровані зовнішніми процесами (довкіллям), є підмножиною $\left(E \times T\right)^*$ з очевидною умовою, що якщо $i = ...\left(e_\alpha, t_\alpha\right)\left(e_\beta, t_\beta\right)...$, то $t_\alpha < t_\beta$. Визначимо $\omega_{t_1}^{t_2}$ як вхідний рядок, який стартує у момент часу t_1 і завершується в момент t_2. Якщо t_2 - кінцеве, то і рядок подій також кінцевий.

Події, як вже наголошувалося, описуються у власній системі відліку. Якщо подія $e_{t_1}^{t_2}$ з'являється в момент t_d за місцевим часом підсистеми, то

вона має бути оброблена в інтервалі $[t_d + t_1, t_d + t_2)$ для досягнення ефекту. Такий же інтервал в *GT* має вигляд $[\tau + t_d + t_1, \tau + t_d + t_2)$. Звичайно, можна дати складнішу інтерпретацію подій, що відбуваються в часі, з «доброю» поведінкою в інтервалі і «поганою» поведінкою поза ним. Якщо часовий інтервал $[0, \infty)$, то тоді подія не прив'язана до часу. Перетинання інтервалів може спростити опис подій. Хай $e_d{}_{t_1}^{t_2}$ і $e_d{}_{t_3}^{t_4}$ - дві часові події для деякого символу d з області D. Якщо справедлива операція $\sigma_{t_1}^{t_2} + \sigma_{t_2}^{t_3}$, то $e_d{}_{\min(t_1,t_3)}^{\max(t_4,t_2)}$ представляє обоє події (рис. 2.3).

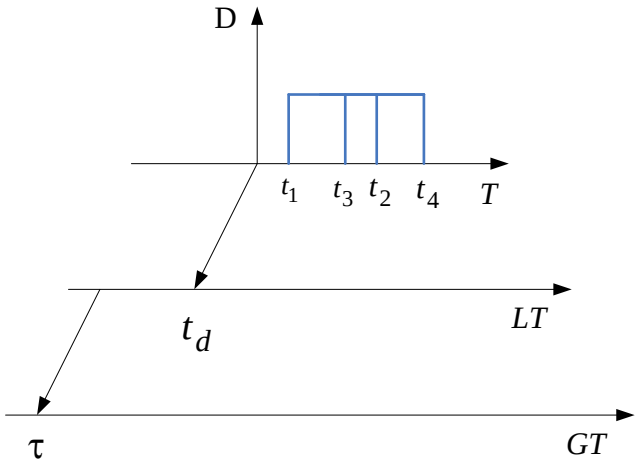


Рис.2.3 Відношення між часовими посиланнями

Множина допустимих вхідних функцій визначена на підмножині *T* і в піддомені *E*. Множина всіх подій, які з'являються на будь-якому кінцевому інтервалі часу, кінцева, тобто має кінцеву потужність. Це вірно, тому що в режимі реального часу система витрачає кінцевий час на обробку будь-якої події. У разі, коли нескінченне число подій може з'являтися на кінцевому інтервалі часу, система не зможе їх обробити за обмежений час. Така поведінка вважається неприйнятною, тому накладається приведена вище умова. Якщо $\omega_{t_0}^{t_1}$ є вхідною функцією, заданою на інтервалі часу з кількістю часових точок *n*, в яких визначена ω, то система оброблятиме

всі входи за період часу n/μ. Ω складається з функцій ω і визначається таким чином: $\Omega = \{\omega : T \to E \mid \omega = e_k(t_{e_k})\}$. Таким чином, будь-яка функція має бути визначена на лічильній множині моментів часу. Якщо $T_d \in T$ - є множина моментів часу, коли визначена функція ω, то існує морфізм $f : T_d \to N$. Якщо $Card(T_d) = \infty$, то f є ізоморфізмом.

Дана умова гарантує, що так званий ефект Zeno [208] на кінцевому інтервалі не з'явиться. Існує і інший тип прояву Zeno. Це відбувається, коли ряд подій, які з'являються на будь-якому кінцевому інтервалі, перевищує кількість місць у вхідній черзі і на обробці. Це пов'язано з обмеженим числом місць в черзі на обробку, з низькою швидкістю обробки подій, з високою інтенсивністю надходження подій в систему [101]. Якщо K - максимальне число подій, які можуть бути буферизовані в системі, то число подій, які можуть з'явитися в інтервалі $\sigma_{t_1}^{t_2}$ повинно бути $N \le K + |t_2 - t_1|\mu$. Якщо кількість подій, що з'являються в період часу більше, ніж $K + |t_2 - t_1|\mu$, то деякі події втрачатимуться. На кінцевому інтервалі часу це означає, що інтенсивність появи подій не повинна перевищувати інтенсивність їх обробки μ.

Вхідні функції або функції обробки вхідних подій, що використовуються, можуть розглядатися як слова деякої мови L_E. Мова визначається на кінцевому алфавіті $E \times T$, де E є набором подій (часових або не часових) і T - це множина часу. Слово в мові w визначається таким чином: $\forall \omega \in \Omega, \forall t \in T,\ w = \omega(t_0)\omega(t_1)...\omega(t_n)$. Мова (L_E) складається з множини всіх слів w таких, що $w = e_0(t_0)e_1(t_1)...e_n(t_n)$ і $t_0 < t_1 < ... < t_n$, де $e_0, e_1, ..., e_n \in E$ і $t_0, t_1, ..., t_n \in T$. Таким чином, L_E є підмножиною $(E \times T)^*$.

2.2.3 Основні визначення дискретних систем реального часу

Моделі систем, які ми називаємо дискретними, мають зчисленну множину станів. Тривалість переходу із стану в стан залежить від типа події, що обробляє система. Таким чином, інтервал (час) переходу задається функцією $\mu : D \to T$. Це означає, що на будь-якому кінцевому інтервалі часу система робить лічильне число кроків різної тривалості.

Визначення 2.2 Дискретна система реального масштабу часу (*ДСРМЧ)* є вісімка

$$A = \left(T, Q, E, Y, \phi, \beta, f_\mu, f_\mu^{\max}\right),$$

де

- $T = R^+$ множина значень часу;

- $E = E^w \bigcup E^{sys} \bigcup E^{timer}$ множина подій і $I = E \times T$ - вхідний алфавіт;

- Q множина символів, відповідних станам;

- $Y = \left(Y_j \middle| j \in J\right)$ індексована множина вихідних величин з $Y_j \in D \times T \times T$;

- $f_\mu = 1/\mu : tran(e) \to T$ - функція переходів;

- $f_\mu^{\max}$ - граничне значення функції переходів $f_\mu \le f_\mu^{\max}$;

- ϕ- функція: $\phi : T \times Q \times I \to Q$, функція наступного стану; якщо система знаходиться в стані q у момент t, то в стані q_1 система буде у момент часу $t + f_\mu\left(tran\left(e_{t_1}^{t_2}\right)\right)$, де $e_{t_1}^{t_2} \in E$ є подія, t_d є момент появи події і $t \ge t_d$ - момент, коли завершується обробка події;

- β - множина загальних функцій $\beta = \{\beta_j \mid \beta_j : Q \to Y_j\}$, які називають вихідними функціями.

Вище приведене визначення не виділяє вихідні і фінальні стани.

Сформулюємо нове визначення, що розділяє ці стани.

Визначення 2.3 [188] *rsДСРМЧ* - це *ДСРМЧ*, в якій об'єднані множини вхідних $q_0 \in Q$ і кінцевих $F \in Q$ станів.

Клас реактивних систем (*reactive systems*) випливає з приведеного вище визначення, якщо множина F є порожньою. Такі системи називатимемо *rsДСРМЧ*.

Робота *ДСРМЧ* зі входами-виходами визначається рядками триплетів (e,t,y), де e - подія, що з'являється відповідно до принципів декомпозиції системи, t - момент, коли відбувається подія, y - вихід системи. Всі запуски системи розглядаються в LT і, отже, стартують в момент $t=0$. Для реактивних систем кількість запусків - нескінченна, а для *ДСРМЧ* - кінцева. Розглянемо *ДСРМВ* A, позначимо Ω_q - множина всіх можливих запусків A, коли система знаходиться в стані q. Для *rsДСРМЧ*, коли вхідний стан відомий, цю множину позначимо Ω. Два стани q_1 і q_2 будуть еквівалентними, якщо $\Omega_{q1} = \Omega_{q2}$. Оскільки *rsДСРМЧ* стартують завжди з одного стану, q_0, називатимемо дві системи *rsДСРМЧ* A і B еквівалентними, якщо $\Omega_A = \Omega_B$.

2.2.4 Композиція і декомпозиція структур

У даному підпункті розглянемо можливість комбінації простих паралельних і послідовних структур, структур із зворотним зв'язком з метою отримання складніших рішень, що відповідають сформульованим вище принципам декомпозиції. У наступних визначеннях встановимо обмеження з метою встановлення морфізму між двома *rsДСРМЧ*.

Визначення 2.4 Нехай $M_1 = (T, E_1, Q_1, Y_1, \delta_1, \beta_1, q_{01})$ і $M_2 = (T, E_2, Q_2, Y_2, \delta_2, \beta_2, q_{02})$ описують дві *rsДСРМЧ*. Морфізм між M_1 і M_2 є множина функцій

$\{tm: T \to T, h: Q_1 \to Q_2, ev: E_1 \to E_2\} \bigcup \{g_j : Y_{1j} \to Y_{2j} \,|\, \forall j \in J\}$, що задовольняють наступним умовам $f(\tau_1) = \tau_2$, $f\left(f_{\mu_1}\right) = f_{\mu_2}$,

$f\left(f_{\mu_1}^{\max}\right) = f_{\mu_2}^{\max}$, $h(q_{01}) = q_{02}$,

$h\left(\delta_1\left(t, q_1, (e_1, t_d)\right)\right) = \delta_2\left(tm(t), h(q_1), \left(ev(e_1), tm(t_d)\right)\right)$ і

$\forall j \in J, g_j\left(\beta_{1j}(q_1)\right) = \beta_{2j}\left(h(q_1)\right)$.

Використання трьох типів композицій - послідовної, паралельної і із зворотним зв'язком, дозволяє отримати з простих структур більш складні. Не дивлячись на велику кількість робіт в області теорії систем, теорія композиції в області побудови формальних моделей представлена слабо [188]. Більшість авторів концентрувалися на фундаментальному описі систем і паралельної композиції [208, 198, 201, 172]. Паралельна композиція забезпечує досить точний опис поведінки для не взаємодіючих систем, що функціонують одночасно (паралельно). Для систем, які взаємодіють одна з одною, необхідні складніші конструкції. Деякі визначення паралельної композиції викладені в [202]. Наступні класичні роботи з теорії систем [123] і теорії автоматів [167] вже містять визначення послідовної, паралельної композиції і композиції із зворотним зв'язком.

2.2.4.1 Послідовна композиція

У послідовній конфігурації (рис. 2.4) два автомати зв'язано так, що виходи одного автомата пов'язані зі входами іншого автомата. Звичайно, з'єднання може бути реалізоване тоді і лише тоді, коли множина $Y_{12} \subset E_{21}$. Без цього з'єднання з метою сумісності, друга система не зможе зробити жодного переходу. У складеній системі всі компоненти стартують в один і той же час τ і, таким чином, множина T є загальною для обох підсистем. Крім того, вихід з першої системи має бути подією, що формує події в другій

підсистемі. Без цього, на виході генеруватиметься незчисленна множина подій.

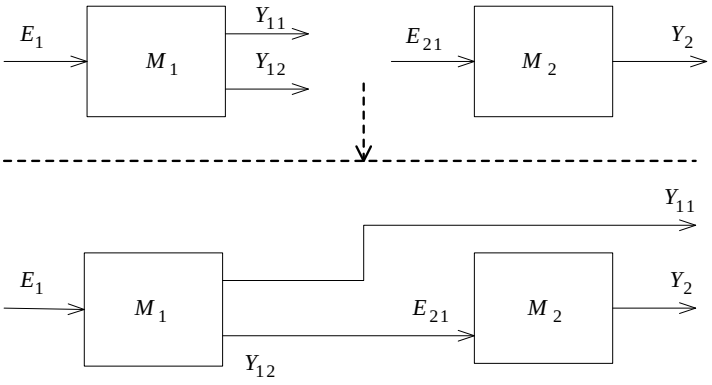


Рис. 2.4 Послідовна композиція *rsДСРМЧ*

Множини допустимих входів, є, як правило, різними в різних *rsДСРМЧ* через різні часи переходів і довжини вхідних черг. Для того, щоб отримати в результаті працездатну систему, необхідно ввести обмеження на допустимі входи для запуску композитної системи. Допустимі запуски мають відповідні часи переходів. Аби отримати їх для композитної системи, необхідно порівняти часи переходів для її частин. Якщо $\frac{1}{\mu_1} \geq \frac{1}{\mu_2}$, то дії, породжені першою системою з'являтимуться з інтервалами $\frac{1}{\mu_1}$, а друга система синхронно оброблятиме їх в процесі появи. Отже, множина прийнятних входів для M_1 є множиною прийнятних входів для композитної системи. Якщо $\frac{1}{\mu_1} < \frac{1}{\mu_2}$, то множина допустимих входів для композитної системи має бути зменшена так, щоб швидкість прибуття подій була менша, ніж μ_2. Тому швидкість прибуття подій в композитній системі повинна перевищувати $\min(\mu_1, \mu_2)$. Ця властивість випливає з особливостей побудови «тандемних» (послідовних) систем обробки подій, що в теорії масового обслуговування пов'язано з теоремами Бйорке і Джексона [101].

Надалі розглядатимемо лише випадки, коли $\mu_1 < \mu_2$. Тоді час переходу композитної системи становитиме $\frac{1}{\mu} = \frac{1}{\mu_1} + \frac{1}{\mu_2}$. Це рівняння стверджує, що виходи змінюються лише після того, як вхідна подія пройде систему. Для *rsДСРМЧ* існує умова, яка може обмежувати довжину L (розмір) послідовної композитної системи. Цю умову можна записати таким чином:

$$(\forall e_{t_1}^{t_2} \in E) \sum_{i=1}^{L} f_{\mu_i} \le f_{\mu_e}^{\max}. \qquad (2.6)$$

Визначення 2.5 Нехай $M_1 = (T, E_1, Q_1, Y_1, \delta_1, \beta_1, q_{01})$ і $M_2 = (T, E_2, Q_2, Y_2, \delta_2, \beta_2, q_{02})$ - дві *rsДСРМЧ*, де $Y_1 = (Y_{11}, Y_{12})$, $\beta_1 = (\beta_{11}, \beta_{12})$. Послідовну композицію M_1 і M_2 позначимо як $(M_1 \Rightarrow M_2)$. Вона описується як

$$M = M_1 \Rightarrow M_2 = (T, E, Q, Y, \delta, \beta, q_0), \qquad (2.7)$$

де

- $Q = Q_1 \times Q_2$ множина станів;
- $E = E_1$ множина подій і $I = E_1 \times T$ вхідний алфавіт;
- $Y = (Y_{11}, Y_2)$ множина виходів;
- $\delta : T \times Q \times I \to Q$ функція наступного стану;

$\delta(t, (q_{01}, q_{02}), (e_1, t_d)) = (\delta_1(t, q_{01}, (\beta_{11}(q_{01}), t_d)), \delta_2(t, q_{02}, (\beta_2(q_{02}), t_d)))$;

- $\beta = (\beta_{11}, \beta_2)$ множина вихідних функцій;
- $q_0 = (q_{01}, q_{02})$ вхідний стан.

Розглянемо випадок, коли декілька входів, застосовуються до тих же систем. Є дві можливі точки зору на ці види структуризації входів. Перша передбачає, що події мають бути синхронізовані [188].

Для пояснення цього розглянемо дві *rsДСРМЧ* M_1 і M_2, що працюють паралельно.

2.2.4.2 Паралельна композиція

Нехай E_1 - множина можливих подій для M_1 і E_2 - для M_2. Тоді, множина подій для результативної системи буде

$$E' = \{(e_1, \varepsilon) | e_1 \in E_1\} \cup \{(\varepsilon, e_2) | e_2 \in E_2\} \cup \{(e_1, e_2) | e_1 \in E_1, e_2 \in E_2\}.$$

Множина подій для паралельної композиції означає, що події можуть з'явитися в один і той же момент часу, тобто синхронно, і можуть з'явитися асинхронно. Для приведення цих ситуацій до одного вигляду введемо порожню подію ε для узагальненого синхронного представлення композиції. Такий підхід розглядає всі події в черзі до одного блоку, як такі, що поступають і обробляються синхронно. Якщо $I_j = E_j \times T$ - це вхідний алфавіт для одного входу, тоді, оскільки $T \times T \times ... T \cong T$, алфавітом для композитних входів буде $E_1 \times E_2 ... E_n \times T$.

Другий підхід передбачає, що входи обробляються незалежно. Проте, така точка зору суперечить нашому припущенню, що система - монолітна.

У паралельній конфігурації множина подій E в результівній машині є підмножиною декартова добутку множин складових подій об'єднуваних машин $E \subset E_1 \times E_2$. Як і в послідовній композиції, в даному випадку обоє підсистеми мають свою систему відліку часу і свій початковий момент часу (час старту). Це гарантує, що елементи декартова добутку відповідають подіям машини, що скомпонована.

Визначення 2.6 Нехай $M_1 = (T, E_1, Q_1, Y_1, \delta_1, \beta_1, q_{01})$ і $M_2 = (T, E_2, Q_2, Y_2, \delta_2, \beta_2, q_{02})$ - дві *rsДСРМЧ* (рис. 1.4) з $E_1 = (E_{11} \times E_{12})$ і $E_2 = (E_{21} \times E_{22})$. Паралельну композицію M_1 і M_2 позначимо $(M_1 \| M_2)$. Вона представляє систему $M = M_1 \| M_2 = (T, E, Q, Y, \delta, \beta, q_0)$,

де

- T множина значень часу;
- $Q = Q_1 \times Q_2$ - множина станів і $q_0 = (q_{01}, q_{02})$;

- $E = (E_{11} \times Z \times E_{22})$ з $Z \subseteq E_{12} \bigcap E_{21}$ - множина подій і $I = E_{11} \times Z \times E_{22} \times T$ - вхідний алфавіт;

- $Y = (Y_1, Y_2)$ множина вихідних значень;

- $\delta : T \times Q \times I \to Q$ функція наступного стану, визначувана як $\delta(t,((q_1,q_2),(e_1,z,e_2,t_d)) = (\delta_1(t,q_1,(e_1,z,t_d)),\delta_2(t,q_2,(e_2,z,t_d)))$;

- $\beta = \{\beta_1, \beta_2\}$ множина вихідних функцій.

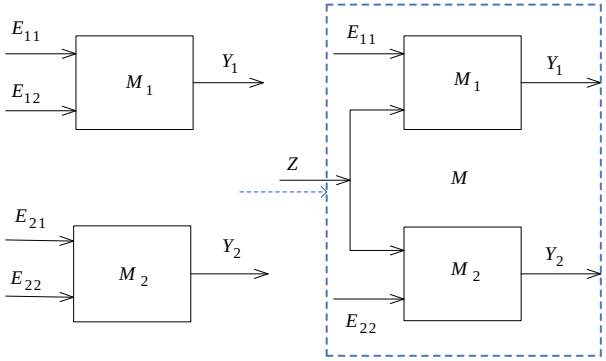


Рис. 2.5 Паралельна композиція *rsДСРМЧ*

Час переходів в композитній системі рівний $\max(f_{\mu_1}, f_{\mu_2})$ і повинен задовольняти умові

$$\max(f_{\mu_1}, f_{\mu_2}) \le \min(f_{\mu_1}^{\max}, f_{\mu_2}^{\max}). \qquad (2.8)$$

В умовах проектування розподілених систем керування сортувальними гірками це жорстке обмеження (2.8) може коригуватися (див. розділ 4).

2.2.4.3 Композиція зі зворотнім зв’язком

Введення зворотних зв'язків в систему, коли виходи системи з'єднуються з її входами, утворює іншу систему. Системи із зворотним зв'язком можуть виконати декілька переходів по одній події на вході, оскільки їх вихід на вході дає нову подію, яка повинна оброблятися. *rsДСРМЧ*, за **визначенням 2.2**, виконують лише один перехід при появі

події. В цьому випадку ми визначимо, що перехід в системі із зворотним зв'язком матиме місце лише тоді, коли система досягне стійкого стану, і знаходитиметься в очікуванні вхідної події. Вочевидь, що система може зробити декілька внутрішніх переходів, аби увійти до такого стану. Множина допустимих функцій входу для системи із зворотним зв'язком - це лише підмножина множини функцій входу у первинній системі. У результаті, зворотний зв'язок призводить до збільшення часу обробки і робить нестійким виконання умови $\max(f_\mu) \le \min(f_\mu^{\max})$.

Визначення 2.7 Нехай $M_1 = (T, E_1, Q_1, Y_1, \delta_1, \beta_1, F_1, q_{01})$ - це *rsДСРМЧ* (рис. 2.6) з $E_1 = (E_{11} \times E_{12})$ і $Y_1 = (Y_{11}, Y_{12})$. Зворотний зв'язок в M_1 утворює систему

$$M = (T, E, Q, Y, \delta, \beta, F, q_0),$$

де

- T множина значень часу;
- $Q = Q_1$ множина станів з $q_0 = q_{01}$;
- $E = E_{11}$ множина подій і $I = E \times T$ - вхідний алфавіт;
- $Y = Y_{11}$ множина виходів;
- $\beta = \{\beta_1\}$ множина вихідних функцій;
- $\delta : T \times Q \times I \to Q$ функція наступного стану:

$$q_1(t + n * f_\mu) = \delta(t, q, (e_{11}, t_d)) = \delta_1(...(\delta_1(t, q, (e_{11}, t_d)))).$$

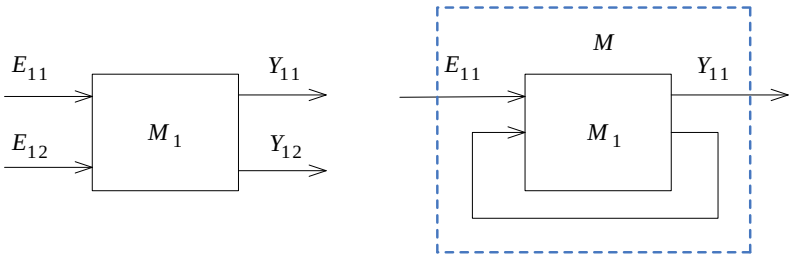


Рис. 2.6 Композиція *rsДСРМЧ* із зворотним зв'язком

Завдяки своїй природі, деякі системи із зворотним зв'язком можуть мати безмежний сумарний час переходів, тому що система виконуватиме нескінченне число внутрішніх переходів. Це означає, що клас всіх *rsДСРМЧ* не вичерпується операцією введення зворотного зв'язку. Застосовувати її необхідно з крайньою обережністю.

2.2.5 Багатосортна алгебра, морфізм і гомоморфізм структур дискретних систем керування

Зв'язок між теорією автоматів і алгеброю, що зараз здається сповна природним, був вперше відмічений Бучем [173], який вивчав автомати як унарну алгебру. Знижуючи клас абстрактних автоматів до унарної алгебри, він обмежував набір структур, що вивчалися. Вказуючи на проблеми, Буч також відзначав, що, застосовуючи загальніший підхід, можна вивчати різні структури. Він запропонував використовувати багатосортну алгебру (multi sorted algebra), яка дає переваги за рахунок індексованих носіїв алгебри. Таким чином, сукупність умов, накладених на оператори алгебри, може бути задовільна на підмножині носіїв. Відмітимо зв'язок універсальної алгебри з іншими областями математичних досліджень, такими як, теорія категорій, теорія автоматів та іншими [168, 166].

Алгебричні методи дослідження можуть допомогти, коли необхідно зробити перехід від автоматів як моделі обчислень до математичної логіки для їх опису [36]. Зв'язок універсальної алгебри з логікою означає, що можна використовувати зв'язок між теорією логіки і абстрактними моделями. Саме з цієї причини доцільний підхід до *ДСРМЧ* і до розподілених систем через алгебричну теорію.

Розглянемо зараз структуру *ДСРМЧ* як універсальну алгебру. Множинна природа структури *ДСРМЧ* робить її природним кандидатом на вивчення з використанням багатосортної алгебри [117].

Нехай $M = (T, Q, E, Y, \delta, \beta)$ буде розподіленою системою з

дискретними подіями. Умова $t \le t_d$ робить δ неповною. Щоб зробити модель системи відповідною для використання структур універсальної алгебри необхідно зробити δ загальною функцією. Це можна зробити, якщо розширити поняття δ таким чином, що для всіх двійок (t,t_d), де функція невизначена, вважатимемо, що $\delta(t,q,(e,t_d)) = q$. У решті випадків δ розглядатиметься як розширена функція наступного стану $\overline{\delta}$. Як вже наголошувалося раніше, для *ДСРМЧ* аналізувався тільки реактивний випадок з $F = \varnothing$, де $F = \{ f \,|\, f : T_d \to N \}$. Отже, множина F не з'являється в явному вигляді. Як і очікувалося, *ДСРМЧ* з розширеною функцією переходів, має структуру універсальної алгебри, що доводиться в наступних теоремах.

Теорема 2.1 Розподілена система з дискретними станами (*РСДС*) є багатосортною алгеброю.

Доведення. Нехай $M = (T, Q, E, Y, \delta, \beta)$ буде *РСДС*. Застосуємо сортування до кожної з множин. Тоді T має сортування s_1 і Q - сортування s_2. Множина вихідних значень має декілька підмножин: $Y = (Y_1, ..., Y_j)$. Кожна підмножина Y_j має сортування *j*. Множина E має сортування *i*. Таким чином, множина сортувань S складається з $\{s_1, s_2, i\} \cup \{ j \,|\, j \in J \}$. Для S розглянемо сигнатуру $\sum = (\delta, q_0, \beta_j \,|\, j \in J)$ з наступними арностями: $ar(\delta) = (s_1 s_1 s_2 i, s_2)$, $ar(\beta_j) = (s_2, j)$. Уточнимо отримання арності для δ. Оскільки δ за визначенням $\delta : T \times Q \times I \to Q$, а $I = E \times T$, то, зробивши підстановку, отримаємо $\delta : T \times Q \times E \times T \to Q$ або в індексах $s_1 s_1 s_2 i, s_2$.

Розглянемо основну несучу множину $C = (T_{s_1}, Q_{s_2}, E_i, \{Y_j \,|\, j \in J\})$ і функцію δ і $\beta_j \in \beta$ з M як операційні символи з $\sum$. Тоді, за визначенням багатосортної алгебри, $A = \left(C, \delta, \left\{ \beta_j \,|\, j \in J \right\} \right)$ є багатосортною алгеброю. Звідси M є багатосортною алгеброю.

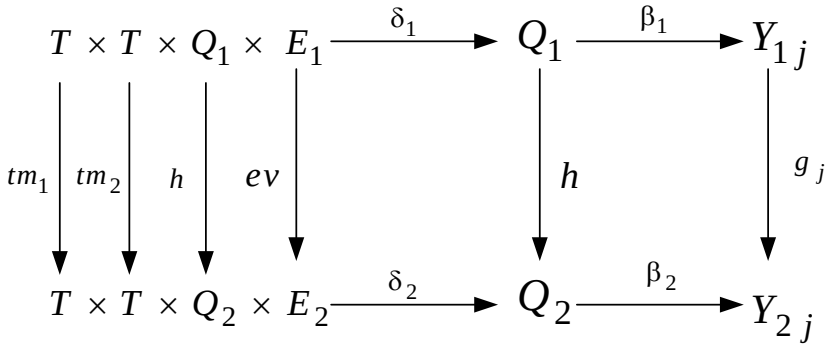
Теорема 2.2 *rsДСРМЧ* - багатосортна алгебра.

Доведення. Нехай $M=(T,Q,E,Y,\delta,\beta,q_0)$ є *ДСРМЧ*. У разі *РСДС* ми розглядали множину $C=(T_{s_1},Q_{s_2},E_i,\{Y_j \mid j\in J\})$ з арностями, які специфікуються індексами.

Якщо розглянутий початковий стан q_0 буде операційним символом, тоді ми отримаємо сигнатуру $\sum=(\delta,\{\beta_j \mid j\in J\},q_0)$. Сигнатура має наступні арності: $ar(\delta)=(s_1s_1s_2i,s_2)$, $ar(\beta_j)=(s_2,j)$, $ar(q_0)=s_2$. Таким чином, множина C з операцією, визначеною у σ формі, є багатосортна алгебра. Отже, M - теж багатосортна алгебра.

Звідси, *ДСРМЧ* - багатосортна алгебра. У цьому контексті цікаво показати, що морфізм між *ДСРМЧ* є гомоморфізмом в царині універсальних багатосортних алгебр.

Слідство 2.1 Морфізм між двома *ДСРМВ* є гомоморфізм між основними алгебрами.



Доведення. Нехай $M_1=(T,E_1,Q_1,Y_1,\delta_1,\beta_1,q_{01})$ і $M_2=(T,E_2,Q_2,Y_2,\delta_2,\beta_2,q_{02})$ - дві *ДСРМЧ* з Y_1 і Y_2, визначеними на деякому J. З теореми 2.2 M_1 і M_2 - алгебри з сігнатурою $\sum=(\delta,\{\beta_j \mid j\in J\},q_0)$.

Множина функцій

$$\{tm_1:T\to T,tm_2:T\to T,h:Q_1\to Q_2,ev:E_1\to E_2\}\bigcup\{g_j:Y_{1j}\to Y_{2j}\mid\forall j\in J\}$$

задовольняє умовам морфізму і, за визначенням 2.4, також задовольняє умовам багатосортного гомоморфізму. У сигнатурі $\sum$ ми маємо:

- $h(q_{01}) = q_{02}$ гіпотези;

- $\forall f_1 \in F_1 \ \exists f_2 \in F_2$ такі, що $h(f_1) = f_2$ - гіпотези;

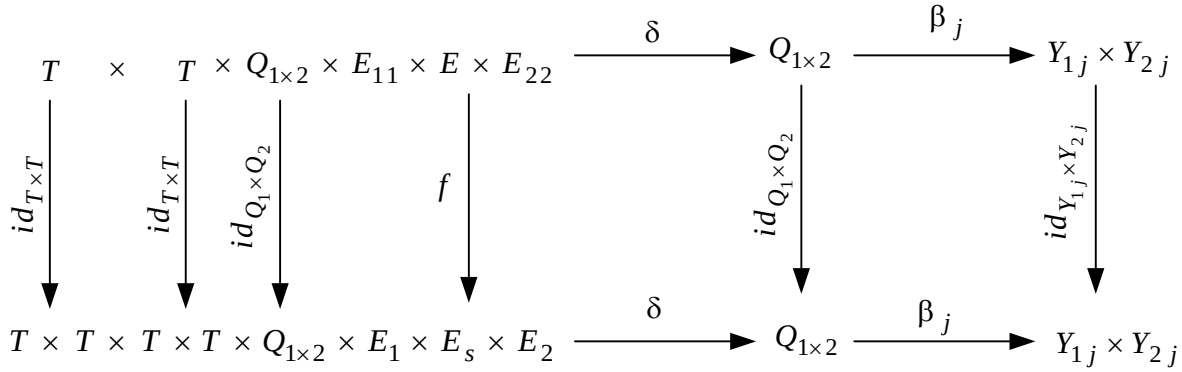
- $h(\delta_1(t_1, t_2, q_1, e_1)) = \delta_2(tm(t_1), tm(t_2), h(q_1), ev(e_1))$;

- $\forall j \in J, g_j(\beta_{1j}(q_1)) = \beta_{2j}(h(q_1))$.

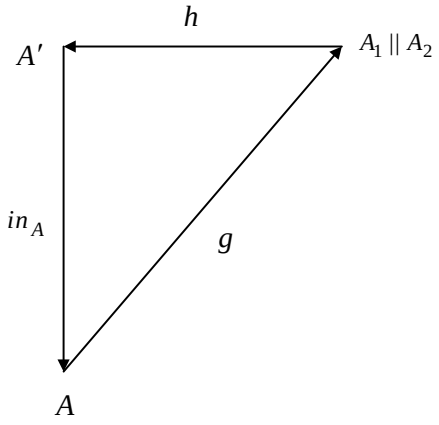
Таким чином, за визначенням морфізмів $\left[tm_1, tm_2, h, ev, (g_j \mid j \in J)\right]$ є $\sum$-гомоморфізм.

Поняття прямого добутку алгебр відображає поведінку паралельно працюючих автоматів. Нехай $A_1 = (T, E_1, Q_1, Y_1, \delta_1, \beta_1, q_{01})$ і $A_2 = (T, E_2, Q_2, Y_2, \delta_2, \beta_2, q_{02})$ - дві *rsДСРМЧ* і B - їх паралельна композиція за визначенням 2.6. Прямий добуток $A = A_1 \times A_2$ є алгебра $A = (T \times T, E_1 \times E_2, Q_1 \times Q_2, Y_1 \times Y_2, \delta, \beta, (q_{01}, q_{02}))$, де функції δ і $\beta_j \in \beta$ визначені раніше. Розглянемо відношення між декартовим добутком двох алгебри і паралельною композицією *rsДСРМЧ*.

Спочатку розглянемо паралельне з'єднання двох *rsДСРМЧ*, як показано на рисунку 2.5. Нехай $B = (T, E, Q, Y, \delta, \beta, q_0)$ - багатосортна алгебра, отримана в результаті паралельної композиції A_1 і A_2. Для паралельного з'єднання, ми повинні накласти умову $E = E_{12} \bigcap E_{21}$. Враховуючи підмножину $E_s = \{(e,e) \mid e \in E\} \subset E_{12} \times E_{21}$ ми можемо побудувати підалгебру A' з A (прямий добуток A_1 і A_2), утворену з тих же множин, що і A, за винятком множини подій, в якій ми братимемо тільки підмножину $E_v = E_{11} \times E_s \times E_{22} \subset E_{11} \times E_{12} \times E_{21} \times E_{22}$. Крім того, розглянемо наступну функцію: $f : E_{11} \times E \times E_{22} \to E_{11} \times E_s \times E_{22}$, яка визначається як $f(e_{11}, e, e_{22}) = (e_{11}, e, e, e_{22})$. Як вже мовилося раніше, множина часу є $T \times T \cong T$. Розглянемо функцію $id_{T \times T} : T \to T \times T$, що визначається як: $id_{T \times T}(t) = (t,t)$. Таким чином, $h = \left[id_{T \times T}, id_{Q_1 \times Q_2}, f, id_{Y_j}\right]$ є гомоморфізмом між B, алгеброю, представленою паралельним з'єднанням, і A'. Позначимо $E_1 = E_{11} \times E_{12}$, $E_2 = E_{21} \times E_{22}$ і $Q_{1\times 2} = Q_1 \times Q_2$.



Отже, A' є підалгеброю A. Тому, існує гомоморфізм $in_A : A' \to A$. Застосовуючи універсальну властивість алгебри ми отримуємо гомоморфізм з B в A. Отже, ми отримуємо гомоморфізм основної алгебри паралельного з'єднання і декартова добутку $A_1 \times A_2$.



Відображення $A_1 \times A_2$ в B також можливе. В цьому випадку нам потрібна функція $tm : T \times T \to T$ і функція: $g : E_{11} \times E_{12} \times E_{21} \times E_{22} \to E_{11} \times E \times E_{22}$. Функція tm може бути визначена так

$$tm(t_1, t_2) = \begin{cases} t, & \text{якщо } t_1 = t_2 \\ 0, & \text{інакше} \end{cases},$$

і функція g з наступним описом:

$$g(e_{11}, e_{12}, e_{21}, e_{22}) = \begin{cases} (e_{11}, e, e_{22}), & \text{якщо } e_{12} = e_{21} \\ (e_{11}, \varepsilon, e_{22}), & \text{інакше} \end{cases}.$$

Таким чином, $k = \left[tm, tm, g, id_{Q_1 \times Q_2}, id_{Y_j} \right]$ є гомоморфізм з $A_1 \times A_2$ у B.

Автоматні моделі обчислювальних процесів наведені в роботі [82].

РОЗДІЛ 3

МЕТОДИ ПОБУДОВИ І ДОСЛІДЖЕННЯ ІНФОРМАЦІЙНИХ, ФУНКЦІОНАЛЬНИХ І ТЕХНІЧНИХ СТРУКТУР ДЕЦЕНТРАЛІЗОВАНИХ АВТОМАТИЗОВАНИХ СИСТЕМ КЕРУВАННЯ

3.1 Принципи побудови М-моделей розподілених інформаційно-керуючих систем

В процесі проектування відповідно до розробленої методики (рис. 5.4), замовник повинен підготувати схеми інформаційних потоків в існуючій і майбутній системі без прив'язки до використовуваних технічних засобів, які підлягають заміні в новій системі. Тобто для замовника і проектувальника ми маємо справу з неіснуючою поки системою, яку назвемо мета системою. Тоді інформаційну модель потоків в ній називатимемо мета моделлю або М-моделлю.

На рис. 3.1 показана інформаційна мета система (ІМС), що взаємодіє з технологічним об'єктом автоматизації (ТОА). ІМС описується інформаційною мета моделлю. У мета моделі об'єкт автоматизації ТОА відображається у вигляді двох не порожніх множин - джерел і приймачів інформації

$I = \{i_1, i_2, ..., i_n\}$, $\Pi = \{\pi_1, \pi_2, ..., \pi_m\}$.

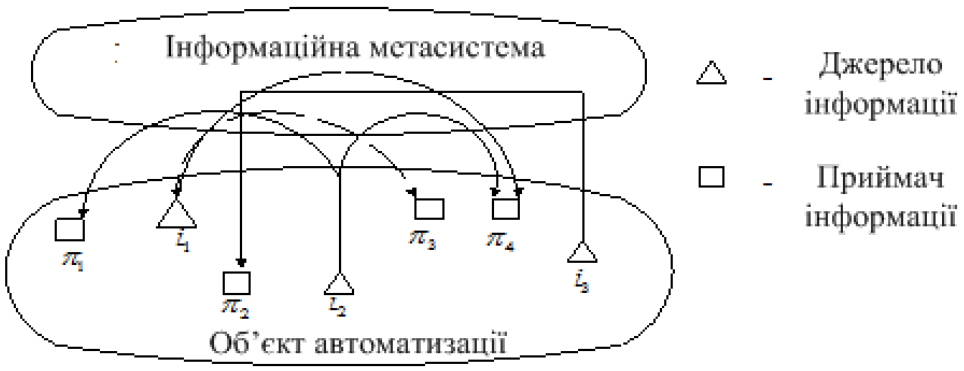


Рис. 3.1 Метамодель системи, що проектується

Кожне джерело i_k може генерувати множину орієнтованих (направлених) інформаційних потоків $P_k = \{p_{k1}, p_{k2}, ..., p_{kl}\}$. Кожен інформаційний потік описується четвіркою:

$$\left\langle OD_{kj},\ b_{kj},\ \lambda_{kj},\ X_{kj} \right\rangle, \tag{3.1}$$

де - $OD_{kj} = (i_k, \pi_g)$ - це OD-пара або Origin-Destination пара, вказуюча джерело i_k і π_g приймач інформаційного потоку;

b_{kj} - середній об'єм повідомлення (інформації), що генерується джерелом i_k, який створює потік p_{kj} (біт); кожне джерело може генерувати декілька потоків, що відрізняються в наступних елементах: π_g (призначення), b_{kj} (об'єм), λ_{kj} (інтенсивність), X_{kj} (характеристики);

λ_{kj} - середня інтенсивність генерації повідомлень в потоці p_{kj} (1/с);

$X_{kj} = \{W_{kj}^P, T_{kj}^{\text{гр}}, K_{kj}^{\text{і}}, K_{kj}^{\text{г}}\}$ - вектор додаткових характеристик потоку p_{kj} : W_{kj}^P - безрозмірний коефіцієнт, що характеризує "вагу" або "важливість потоку", $T_{kj}^{\text{гр}}$ - граничний час затримки обробки і передачі повідомлення b_{kj}, $K_{kj}^{\text{і}}$ - коефіцієнт допустимого спотворення (втрат) інформації в повідомленні при його передачі/обробці $(0 \le K_{kj}^{\text{і}} < 1)$, $K_{kj}^{\text{г}}$ - допустимий коефіцієнт готовності мета системи при передачі/обробці потоку $(0 < K_{kj}^{\text{г}} \le 1)$.

X_{kj} включає характеристики, які вимагають додаткових досліджень ТОА і роботи з експертами, і які не завжди вдається отримати при проектуванні реальних систем.

Очевидно (див. рис. 3.1), що кожне джерело може генерувати декілька потоків, призначених для одного або декількох приймачів. Приймачі ж можуть приймати декілька потоків від одного або декількох джерел. Припускатимемо, що в ІМС процес обробки не впливає на динаміку

потоків (див. рис. 3.2). Тобто, є мета ресурси (гіпотетичні ресурси обробки даних), достатні для реалізації будь-яких функцій обробки.

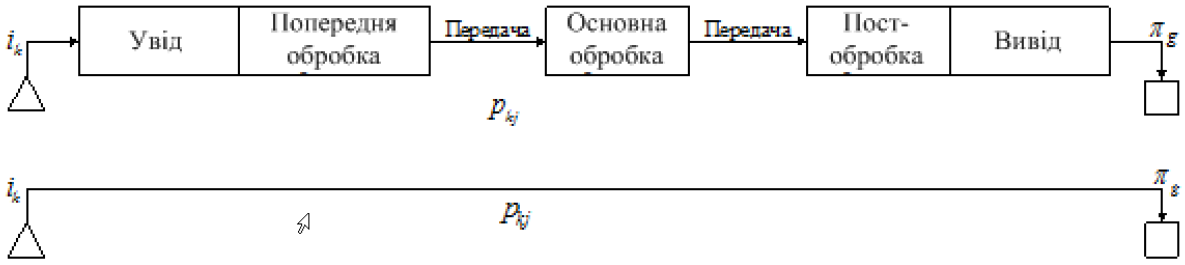


Рис. 3.2 Мета-ресурси

На даному інформаційному рівні не вирішуються завдання, де знаходитимуться пристрої обробки даних (у точці i_k або в точці π_g) і яку вони повинні мати обчислювальну потужність. Вони розглядаються на інших етапах методики (рис. 5.4) [80].

Таким чином, початкову мета модель системи можна представити у вигляді орієнтованого незв'язного мультиграфа з поміченими ребрами (див. рис. 3.3а). Оскільки на реальних ТОА джерела і приймачі можуть знаходитися територіально в одному місці (кімнаті, кросовій шафі і т.п.), то в початковому мультиграфі зведемо всі такі точки до одного елементу, який називатимемо **агрегованим вузлом** (див. рис. 3.3б, рис. 3.3в). Отримана в цьому випадку мета модель складатиметься з множини вузлів. Уточнимо: **вузол** - технологічна підсистема (ТПС), в якій знаходиться **джерело** і/або **приймач** і/або яка додатково може бути **транзитом.** Останній тип вузла може з'явитися в процесі структурної оптимізації. Він відповідає підсистемі, яка є не тільки джерелом і/або приймачем, але і проміжним вузлом в передачі інформаційних потоків. Таким чином множину вузлів (вершин) мета моделі (графа) можна описати як

$$U = \{u_1, u_2, ...u_{N_U}\} = I \cup \Pi \cup U_a \setminus I_a \cup \Pi_a, \qquad (3.2)$$

де U_a - множина вершин, отриманих в результаті агрегації;

I_a, Π_a - множина поглинених (агрегованих) джерел і/або приймачів.

Для аналізу і синтезу інформаційної структури і оптимального перерозподілу інформаційних потоків, введемо два нові поняття: **лінк** і **роут**.

Лінк - це двонаправлений (дуплексний) інформаційний мета канал, що сполучає два вузли, між якими передаються інформаційні потоки. Фактично наявність лінка в моделі визначається тим, чи існує інформаційний потік між його вузлами. У мета моделі множину лінків можна задати таким чином:

$$L^0 = \{l_j\},\ j = \overline{1, N_L},\ N_L = \left|L^0\right|, \tag{3.3}$$

$$l_j = \{(u_{j1}, u_{j2}), P_j, Y_j\}, \tag{3.4}$$

де (u_{j1}, u_{j2}) - вузли (вершини), які сполучає мета канал або лінк;

$P_j = \{\overline{p}_{j1}, \overline{p}_{j2}, ..., \overline{p}_{jk}\}$ - множина потоків роутів, що проходять по *j*-му лінку в двох напрямах;

$Y_j = \{\overleftarrow{C}_j^{\text{гр}}, \vec{C}_j^{\text{гр}}, K_j^{\text{пом}}, K_j^{\text{г}}, W_j^L\}$ - множина додаткових характеристик мета каналу, що відображають особливості його можливої технічної реалізації;

$\overleftarrow{C}_j^{\text{гр}}, \vec{C}_j^{\text{гр}}$ - граничні значення пропускної спроможності лінка в двох напрямах; очевидно, що повинні виконуватися обмеження:

$$\overleftarrow{C}_j^{\text{гр}} \geq \sum_{p_{ji} \in P_j} \overleftarrow{p}_{ji}, \tag{3.5}$$

$$\vec{C}_j^{\text{гр}} \geq \sum_{p_{ji} \in P_j} \vec{p}_{ji}, \tag{3.6}$$

$K_j^{\text{пом}}$ - коефіцієнт появи помилок при передачі повідомлень по мета каналу (пом/біт);

$K_j^{\text{г}}$ - коефіцієнт готовності каналу ($0 < K_j^{\text{г}} \leq 1$);

W_j^L - ваговий коефіцієнт (“вага”, “вартість”) мета каналу.

Вектор характеристик Y_j, як і X_{kj}, вимагає додаткових досліджень ТОА і знання особливостей технічних засобів для побудови керуючого обчислювального комплексу (КеОК). Тому в повному об'ємі Y_j можуть застосовуватися при ітераційному проектуванні систем за розробленою методикою.

Роут - це інформаційний потік, що описується четвіркою (3.1) і доповнений множиною транзитів. У мета системі функціонує множина роутів:

$$R^0 = \{R_1, R_2, ..., R_{N_R}\}, \tag{3.7}$$

де, згідно (3.1),

$$R_i = \{O_i, D_i, b_i, \lambda_i, X_i, T_i\} = \{OD_i, \bar{P}_i, T_i^{(u)}, X_i\}; \tag{3.8}$$

$\bar{P}_i = b_i \times \lambda_i$ - інформаційний потік (біт/с);

$T_i^{(u)} = \{u_{i1}, u_{i2}, ..., u_{iN_T}\}$ - множина транзитних вузлів (не включають OD-пару), по яких передається інформаційний потік; у початковій моделі $T_i^{(u)} = \varnothing$; замість $T_i^{(u)}$ можна використовувати $T_i^{(l)} = \{l_{i1}, l_{i2}, ..., l_{iN_T+1}\}$ - множина транзитних лінків, тоді в початковій моделі $|T_i^{(l)}| = 1$.

Вочевидь, що $N_R \geq N_L$.

Таким чином повна початкова мета модель інформаційної системи з урахуванням (3.2) - (3.8) може бути представлена у вигляді неорієнтованого реброво-зваженого графа з орієнтованими роутами (див. рис. 3.3в)

$$G = (U, L^0, R^0), \tag{3.9}$$

Слід зазначити, що початковий граф може бути незв'язаним, що говорить про наявність в ІМС автономних підсистем. В цьому випадку структурна оптимізація або виконується в межах однієї підсистеми, або в завданні повинні вводитися додаткові умови появи нових лінків і побудови

зв'язного графа (див. рис. 3.3г). Надалі припускатимемо, що граф мета моделі - зв'язний.

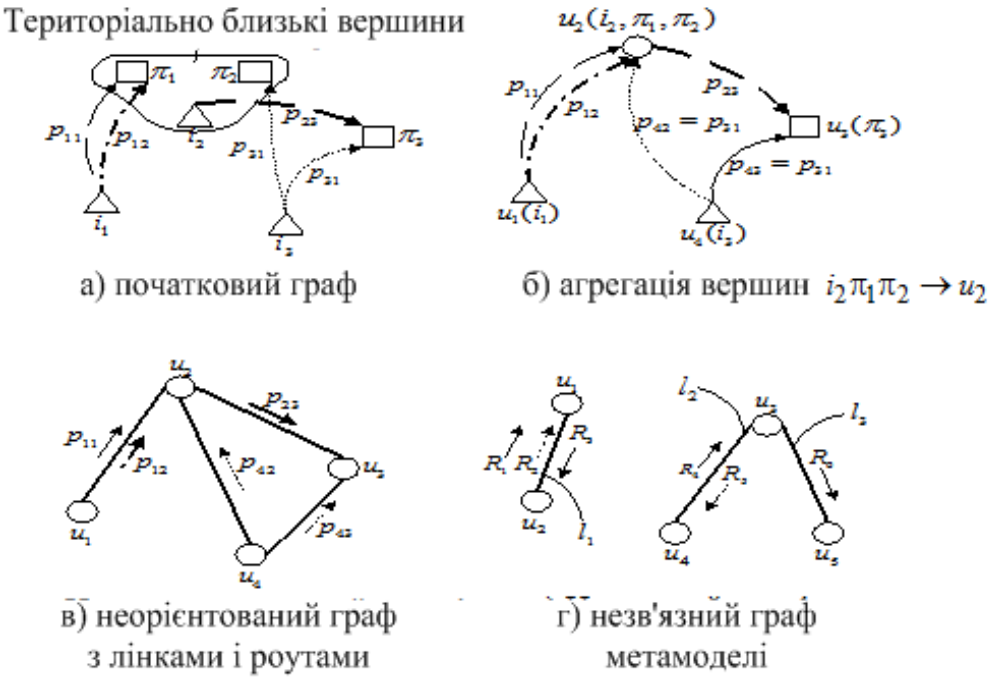


Рис. 3.3 Перетворення М-моделі

3.2 Завдання структурної оптимізації М-моделі і загальний алгоритм його вирішення

У роботі [81] поставлене і вирішене завдання синтезу структури системи на початковому графові (3.9) як пошук такого мінімально зв'язного графа $G^* = (U, L^*, R^*)$, у якого:

$$\{\sum_{R_i^* \in R^*} p_i \times (\left|T_i^{(u)*}\right| + 1) + \sum_{R_j \in R^*} p_j\} - \sum_{R_i \in R^0} p_i \to \min, \qquad (3.10)$$

$$|L^*| \to \min. \qquad (3.11)$$

При наступних “послаблених” обмеженнях:

- $L^* \subset L^0$ - пошук проводиться на початковій множині лінків; лінки можуть видалятися, але не додаватися, в графі;

- $R^* = \{R_i^*\}_{i=\overline{1,N_i}} \cup \{R_j\}_{j=\overline{1,N_j}}$ і $N_i + N_j = N_R$ роути не розщеплюються; тут R_i^* – роут, у якого $T_i^{(u)*} \neq \varnothing$ або $T_i^{(l)*} > 1$;

- роути не утворюють циклів $N_{T_i^{(u)*}} \leq N_U - 2, \ \forall i = \overline{1, N_i}$.

Фактично завдання полягає в пошуку такого мінімально зв'язного графа шляхом видалення лінків і перерозподілі потоків, в якому збільшення сумарного потоку в графі буде мінімальним.

Сформульований критерій може мати інваріантні форми запису залежно від способу опису транзитів ($T_i^{(u)*}$ або $T_i^{(l)*}$). Якщо в (3.10) перейти від $T_i^{(u)*}$ до $T_i^{(l)*}$, то еквівалентний критерій оптимізації матиме вигляд:

$$\sum_{i=1}^{N_R} p_i \times \left|T_i^{(l)*}\right| - \sum_{i=1}^{N_R} p_i \to \min.$$

Виносячи за дужки загальний множник, отримаємо:

$$\sum_{i=1}^{N_R} p_i \times (\left|T_i^{(l)*}\right| - 1) \to \min. \qquad (3.12)$$

В цьому випадку обчислення ведуться по всіх роутах без розбиття на R_j і R_i^* (нерозподілені і розподілені роути), що декілька спрощує алгоритм пошуку оптимальної структури.

Критерії (3.10) і (3.12) припускають використання параметрів роутів p_i і $T_i^{(u)*}$, $T_i^{(l)*}$. Проте критерій має ще одну еквівалентну форму запису через характеристики лінків, опис яких (3.4) включає і направлені потоки, що проходять по них. Тоді мінімізацію приросту сумарного потоку в графі можна записати так

$$\sum_{l_j \in L^*} C_j^* - \sum_{l_j \in L^0} C_j \to \min, \qquad (3.13)$$

$$\sum_{l_j \in L^*} C_j \to \min. \qquad (3.14)$$

З (3.4)-(3.6) бачимо, що $C_j = \vec{C}_j + \overleftarrow{C}_j = \sum_{i=1}^{k} p_{ji}$. При появі у лінку нових потоків, P_j доповнюється новими елементами і k зростає.

Таким чином, в даній постановці завдання з "послабленими" обмеженнями є можливість вибору критерію оптимізації з отриманих варіантів (3.10), (3.12) або (3.14) з урахуванням особливостей побудови структур даних і алгоритмів пошуку рішень.

Розглянемо один з алгоритмів рішення задачі, заснований на імітації видалення найменш завантаженого лінка ($C_j \to \min$) без порушення зв'язності графа і з перерозподілом його роутів по найкоротших шляхах. Узагальнений алгоритм рішення задачі можна представити у вигляді наступної послідовності.

1. Сформувати $L^{(\text{вил})}$ (вибрати лінки, вилучення яких не порушує зв'язності)

2. Доки $L^{(\text{вил})} \neq \varnothing$ виконувати:

2.1 Упорядкувати лінки за C_j

2.2 $\forall l_i \in L^{(\text{вил})}$

2.2.1 Імітація вилучення l_i (з пошуком l_{opt} з мінімальним приростом (3.14))

2.2.2 $\forall p_j \in P_i$ $(\forall R_j \in R_i)$

2.2.2.1 Пошук найкоротшого маршруту (з урахуванням T_j)

2.2.2.2 Перерозподіл роутів по нових маршрутах

2.2.3 Обчислення $\sum_{l_q \in L^*} C_q$

2.2.4 Відновлення l_i і усіх $R_j \in R_i$

2.3 Визначення l_{opt}, у якого $\sum_{l_q \in L^*} C_q \to \min$

2.4 Вилучення l_{opt}, перерозподіл його роутів по найкоротших маршрутах, формування T_i^*

2.5 Обчислення загального критерію відносно початкового стану.

2.6 Сформувати $L^{(вил)}$

3. Кінець.

3.3 Принципи застосування М-моделей в процесі проектування інформаційно-плануючої системи сортувальної станції

Запропоновані М-моделі і завдання структурної оптимізації інформаційно-керуючих систем неодноразово застосовувалися у складі інженерної методики системного проектування при розробці АСК організаційно-технологічними процесами на сортувальних станціях залізниць Росії та України.

Розглянемо методику застосування М-моделей на прикладі проектування інформаційно-плануючої системи (ІПС СС) сортувальної станції "О".

Інформаційно-плануюча система (ІПС СС) є складовою частиною інформаційно-керуючої системи сортувальної станції (ІКС СС). Вона розроблялась на початку 90-х років відповідно до плану науково-дослідних робіт за наказом міністра шляхів сполучення. Замовник ІКС СС - Головне управління руху МШС.

Об'єкт автоматизації. У складі ІПС СС автоматизації підлягають процеси, що відносяться до організаційного управління станцій: обробка даних про вагони і потяги, ведення по вагонних моделей поточного стану станції, автоматизація функцій технічної і товарної контор, довідково-інформаційне обслуговування користувачів, ведення архівних даних, автоматизація звітності аналізів, планування станційних операцій з доведенням розроблених і узгоджених диспетчерським апаратом станції

планів до тих, що входять в ІКС СС АСК технологічними процесами роботи сортувальної гірки і парків станцій.

Особливості станції, що визначають вимоги до ІПС СС.

ІПС СС має бути типовою, яка адаптується до особливостей конкретної станції, що визначається:

а) кількістю сортувальних систем (одна або дві);

б) розвитком шляхів сортувальних систем (паралельне, паралельно-послідовне, послідовне розташування парків; можливість наявності декількох приймально-відправних парків; різне число шляхів в кожному приймально-відправному парку - до 30 при корисній довжині до 150 умовних вагонів; різне число шляхів в сортувальних парках - до 64 при корисній довжині шляхів до 80 умовних вагонів); конфігурацією підходів до станції (до 10) з розмірами руху від 70 до 200 пар вантажних і до 50 пар пасажирських потягів;

в) об'ємом роботи станції (вагонообіг від 10000 до 30000 вагонів за добу, розформування-формування від 3000 до 8000 вагонів в сортувальній системі, вантаження-вивантаження на станції і під'їзних колій до 300 вагонів за добу, робочий парк - до 4000 вагонів в сортувальній системі), тощо.

Методика складається з наступних етапів.

Етап 1. Функціонально-інформаційні зв'язки в ІПС СС і з АСК технологічними процесами роботи сортувальної гірки описується у вигляді таблиць, яки розбиті, для зручності роботи, на окремі підсистеми, виділені фахівцями. Приклад такої таблиці для управління роботою станції в цілому наведений в таблиці 3.1. Повний обсяг всіх таблиць - до 25 сторінок А4. Повний набір таблиць в роботі [1].

Таблиця 3.1

Характеристика потоків інформації, що циркулює між функціональними підсистемами ІПС СС

Найменування підсистеми-джерела інформації	Найменування масиву, що передається	Найменування підсистеми-приймача інформації	Об'єм масиву, що передається (байт)	Кількість передач за добу (разів)	Об'єкт керування
1	2	3	4	5	6
АСОУП	$М^{ПЛ.}$ $М^{ПР.ПР.}$ $М^{ТНЛ}$	УРС УМР ІБД	300 800 2500	8 8 200	станція
АСК відділення дороги (АСК-НОД)	$М^{ПРП}$ $М^{ЗД}$ $М^{ОРГ}$ $М^{ПМУ}$	УРС	192 136 908 690	16 1 2 2	станція
АСКТП парка прибуття або відправлення	$М^{СМД}$	ІБД	14	7375	1 парк
АСКТП гірки (АСКТПГ)	$М^{ТС, СП}$ $М^{СР}$ $М^{ХР}$	ІБД	448 10 200	150 350 150	2 СП 1 гірка
АСК локомотивного депо (АСКЛД)	$М^{ОЛ}$	УСР	4	40	2 системи

Продовження табл. 3.1

1	2	3	4	5	6
УРС	$М^{ПО}$	УПМР	300	16	ПО – 2 шт.
	$М^{ПО}$	АСОУП	300	8	станція
	$М^{ПУП}$	УПМРП	24	16	ПП – 2 шт.
	$М^{ПУП}$	УПМРП	24	16	ПО – 2 шт.
	$М^{Р}$	УСР	6	16	2 системи
	М	АСКЛД	60	2	станція
	$М^{ЗЛ}$	АСКЛД	300	8	станція
	$М^{ППТ}$	УСР	2000	16	2 системи
	$М^{ПО}$	УМР	300	16	станція
	$М^{ПО}$	УСР	300	16	2 системи
УСР	$М^{ПМ}$	УПМР	240	40	ПО – 2 шт.
	$М^{ПП}$	УПМР	72	40	ПО – 2 шт.
	$М^{ВПР}$	УПМР	140	40	ПО – 2 шт.
	$М^{ПР}$	АСКТПГ	140	40	2 гірки
	$М^{СР}$	АСКТПГ	210	40	2 гірки
УПМР	$М^{ГК}$	ПКО	24	82	4 парка
	$М^{ГТ}$	ПТО	24	82	
	$М_{з}^{Л}$	АСКЛД	18	82	
ІБД	$М^{ТС.\ СП}$	УПМР	448	41	СП – 2 шт.
	$М^{ТС.\ ПП}$	УПМР	200	41	ПП – 2 шт.
	$М^{ТС.\ ПО}$	УПМР	200	41	ПО
	$М^{ТНЛ}$	АСОУП	2500	200	
	$М^{ХО}$	УСР	1040	16	2 системи
	$М^{ТСП}$	УСР	72	40	2 системи
	$М^{ТС.\ СП}$	УСР	448	40-120	2 системи
	$М^{ТС.\ ПП}$	УСР	352	40-120	2 системи
	$М^{ДЛИН}$	УСР	200	12	2 системи
	$М^{ТСВП}$	УРС	136	16	2 системи
	$М^{К}$	АСКТП	6	7375	2 системи

Етап 2. За підготовленими таблицями будуються М-моделі для кожної з підсистем. Приклад такої М-моделі наведений на рис. 3.4, 3.5 (повний набір - 38 М-моделей для станції "О" наведено у додатку Ж). Доповнюється набір М-моделей моделлю зовнішніх потоків (рис. 3.6).

Етап 3. Об'єднання всіх М-моделей в загальну модель і використання алгоритму її мінімізації дозволяє отримати підсумкову М-модель всієї системи ІПС СС (рис. 3.7).

Виконані етапи методики роботи з М-моделями відносяться до блоку 15 науково-методичного комплексу КСІ (рис. 5.4). Зрозуміло, що цей варіант структури ще повинен пройти перевірку по інших критеріях, описаних в п.5.4.

Слід зауважити, що обробка М-моделей з використанням алгоритму, описаному у п. 3.2 потребує багато машинного часу в декілька десятків годин, що підтверджує його програмна реалізація OPTiFLOW (п. 3.2). Тому в КСІ запропоновані два метода пошуку раціональних структур М-моделей з використанням генетичних алгоритмів.

3.4 Методи пошуку раціональних варіантів структур інформаційних систем на основі генетичних алгоритмів

3.4.1 Метод раціонального розподілу інформаційних потоків в інформаційно-керуючих системах

Як вже зазначалося в п. 3.2, досліджуваний об’єкт оптимізації у вихідному стані представляється у вигляді множини вузлів типу джерела та приймачі, та множини безтранзитних роутів. Оскільки деякі вузли можуть бути одночасно джерелами та приймачами, або одне джерело може мати декілька приймачів, вихідну структуру можна представити у вигляді орієнтованого реберно-зваженого мультиграфа. В даній частині розглянемо генетичний підхід до пошуку раціональних варіантів інформаційних структур, який відрізняється використанням

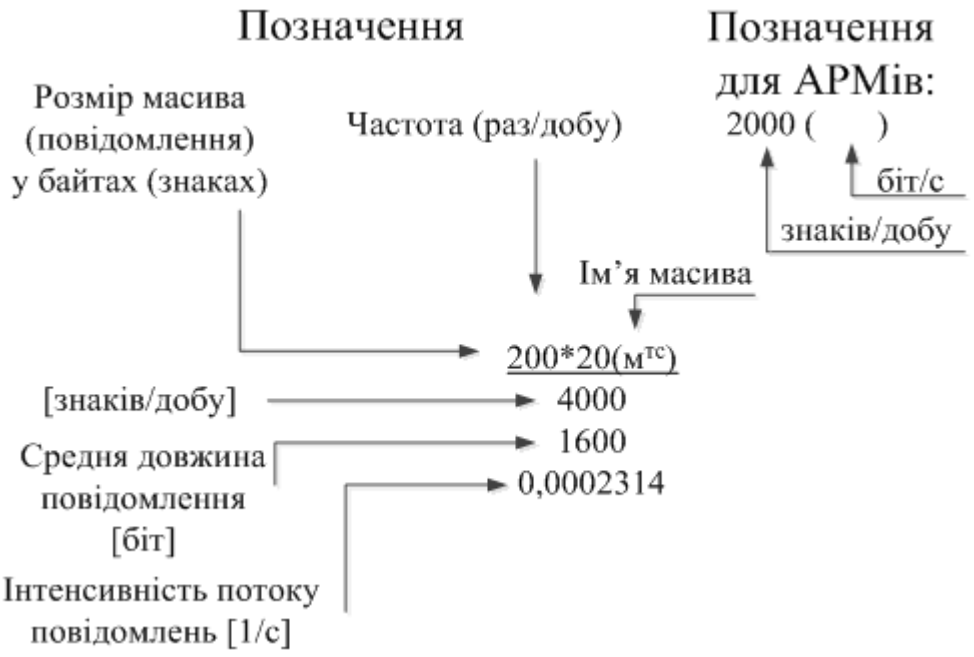


Рис. 3.4 Позначення на М-моделях

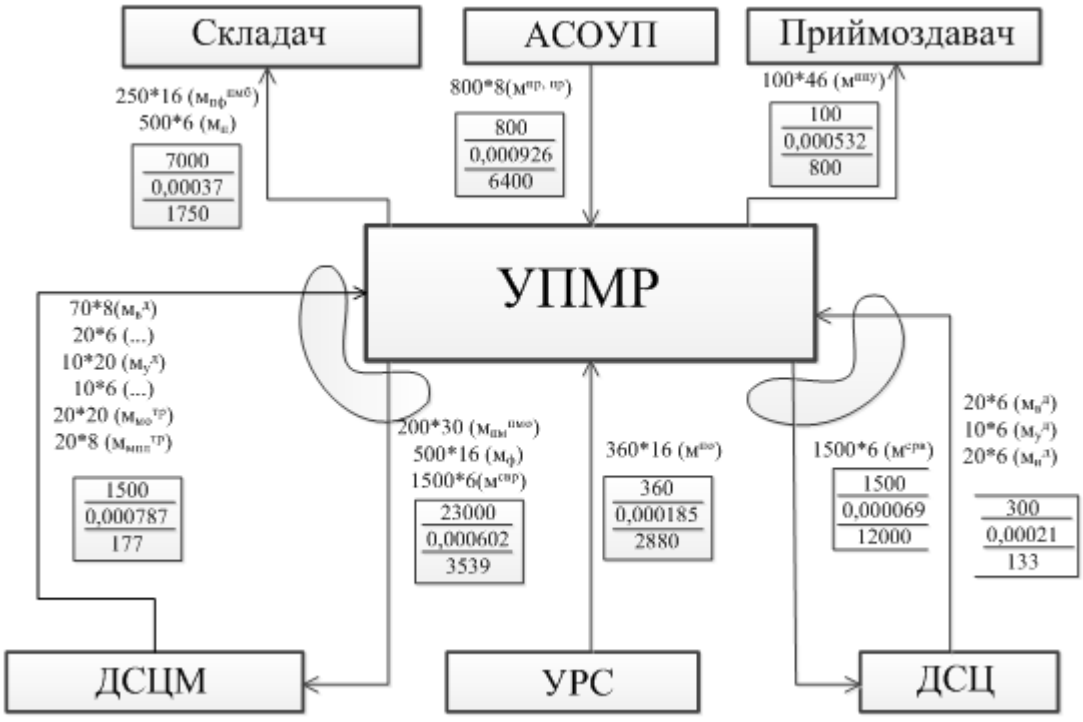


Рис. 3.5 Агрегована М-модель підсистеми управління поїзною та маневровою роботою

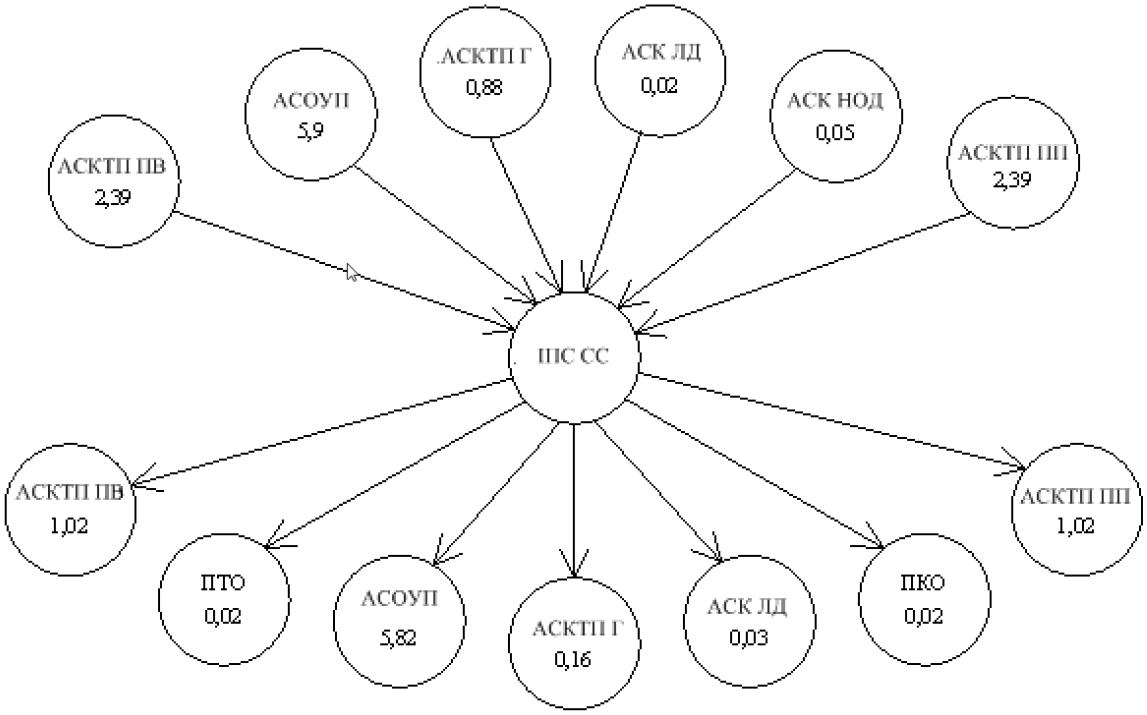
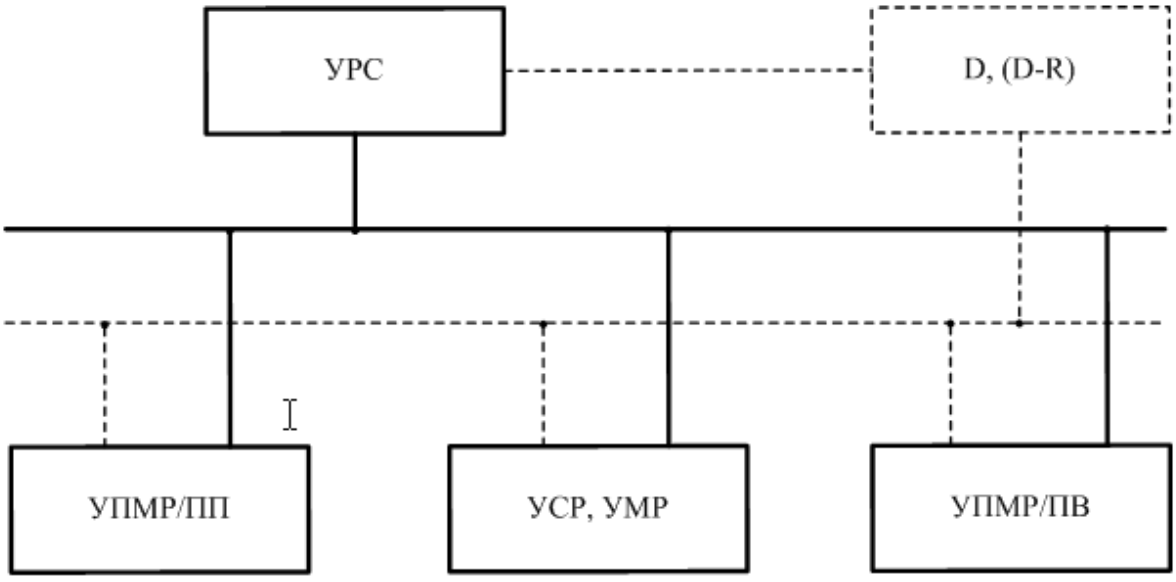


Рис. 3.6 Агрегована М-модель зовнішніх потоків ІПС СС



УРС – управління роботою станції;
УСР – управління сортувальною роботою;
УМР – управління місцевою роботою;
УПМР/ПП – управління поїзною і маневровою роботою в парці прибуття;
УПМР/ПВ – управління поїзною і маневровою роботою в парці відправленя.

Рис. 3.7 Структура ієрархічної децентралізованої системи ІПС СС

інтегрального критерію оптимізації, який поєдную задачі блоків 11 і 15 (рис. 5.4) і формує раціональний варіант з урахуванням сумарної довжини каналів і приросту сумарних інформаційних потоків, що в них передаються [115].

Введемо позначення з уточненнями:

$G = (U, L^0, R^0)$- вихідний орієнтований реберно-зважений мультиграф;

$U = \{u_1, u_2, ..., u_{n_U}\}$- множина вузлів;

$L^0 = \{l_j\}, j = 1,2,..., N_L$ - множина лінків,

де $l_j = \{(u_{j_1}, u_{j_2}), c_j, len\}, a\ c_j = \vec{c}_j + \overleftarrow{c}_j$;

$R^0 = \{r_1, r_2, ..., r_{N_R}\}$- множина роутів, де

$r_i = \{o_i, d_i, p_i, T_i\}$- і-й роут, в якому o_i,d_i - пара OD-вузлів або OD-пара (Origin – Destination), а p_i – потік інформації між вузлами OD-пари;

$T_i = \{u_{i_1}, u_{i_2}, ..., u_{N_T}\}$- множина транзитів.

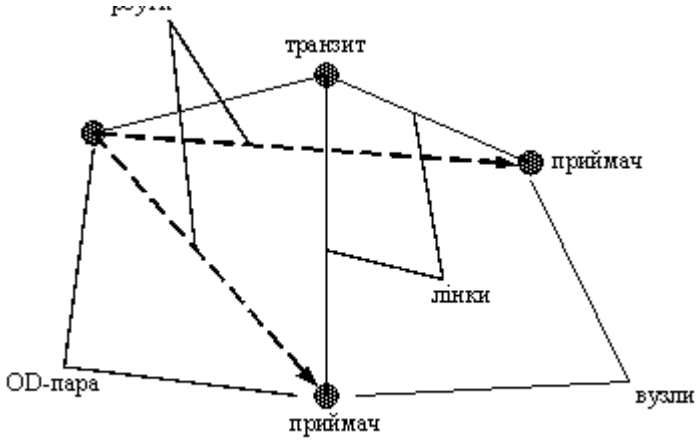


Рис. 3.8 Початковий граф для оптимізації

На основі введених визначень та позначень задача оптимізації розподілу інформаційних потоків в системі, що автоматизується, формулюється наступним чином.

Для заданого орієнтованого реберно-зваженого мультиграфа

$G=\left(U,L^0,R^0\right)$, у якого $\forall i=1,2,...,N_R$, $T_i=\varnothing$ та $N_R\geq N_L$, знайти такий мінімально зв'язний граф $G^*=\left(U,L^*,R^*\right)$, щоб

$$Pk=\left[\sum_{r_i^*\in R^*}p_i\times\left(\left|T_i^*\right|+1\right)+\sum_{r_j^*\in R^*}p_j\right]-\sum_{r_i\in R^0}p_i\to\min \qquad (3.16)$$

При цьому задані наступні обмеження:

$R^*=\left\{r_i^*\right\}_{i=\overline{1,N_i}}$ та $N_i+N_j=N_R$ (роути не розчіплюються),

де $r_i^*=\left\{o_i,d_i,p_i,T_i^*\right\}$, $T_i^*\neq\varnothing$ (потоки не змінюються за величиною), причому $N_{T_i^*}\leq N_U-2,\ \forall i=\overline{1,N_i}$ (роути не створюють циклів).

Також можна ввести критерій мінімізації по сумарній довжині лінків:

$$Lk=\sum_{len_i^*\subset L^*}len_i\to\min. \qquad (3.17)$$

Остаточний критерій для визначення оптимальності розв'язку виглядає наступним чином:

$$K=\left(C_1\cdot Pk+C_2\cdot Lk\right)\to\min. \qquad (3.18)$$

Описаний набір обмежень будемо називати „послабленими обмеженнями", оскільки вони не враховують можливі технічні обмеження на значення c_j (або на окремі складові).

Запропонований критерій (3.16) синтезу структури, що містить мінімальну кількість лінків якомога коротшої довжини, які забезпечують передачу усіх роутів та володіють мінімальним сумарним приростом інформаційних потоків, забезпечує отримання і більш високих показників надійності реалізації роутів. Це пов'язано з тим, що вираз (3.16) забезпечує мінімальну довжину ланцюжків проходження роутів.

Алгоритм розв'язання задачі оптимального розподілу інформаційних потоків заснований на імітації видалення лінків і перерозподілі роутів з відкатом і оцінці сумарного приросту інформаційних потоків для кожного

лінка – кандидата на видалення. Для скорочення часу пошуку оптимального варіанта використовуються процедури впорядкування та направленого перебору.

В даному методі, як базовий, використовується генетичний алгоритм. Окрім цього при розв'язку використовується декілька „класичних" алгоритмів з теорії графів: перевірка графа на зв'язність та знаходження маршрутів проходження роутів.

Даний алгоритм відрізняється від запропонованого в п. 3.2 [80], насамперед, тим, що обмеження на пошук розв'язків лише на множині L^0 не накладається. Тобто в структурному графі системи можуть бути присутніми лінки, що не входять до заданої умовою первинної множини лінків L^0.

Проектування хромосоми. Як відомо, проектування хромосоми – складна і відповідальна справа, адже від її побудови залежить швидкість пошуку розв'язку.

В даній задачі хромосома кодує розв'язок у вигляді квадратної матриці суміжності, що розгорнута в одну стрічку. Матриця суміжності *A* (рис. 3.8) представляє собою квадратну матрицю *n* x *n*, що складається з „0" та „1", де *n* – кількість вершин графу (вузлів інформаційно-керуючої системи - ІКОС). Елемент a_{ij} вказує на наявність дуги (лінку) між вершинами *i* та *j*.

$$A = \begin{bmatrix} a_{11} & a_{12} & \dots & a_{1n-1} & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n-1} & a_{2n} \\ \dots & \dots & \dots & \dots & \dots \\ a_{n-1,1} & a_{n-1,2} & \dots & a_{n-1,n-1} & a_{n-1,n} \\ a_{n1} & a_{n2} & \dots & a_{n,n-1} & a_{nn} \end{bmatrix} \quad a_{ij} = \begin{cases} 1 & \text{, якщо є лінк } l_{ij} \\ 0 & \text{, якщо немає лінку } l_{ij} \end{cases}$$

Рис. 3.9 Матриця суміжності

Таким чином хромосома виглядала б так (рисунок 3.10).

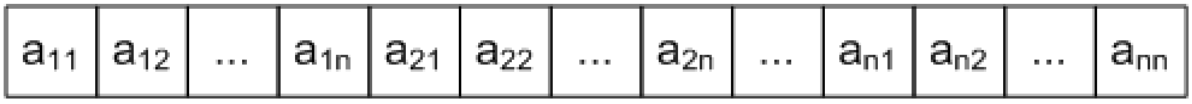


Рис. 3.10 Вигляд не оптимізованої хромосоми

Довжина хромосоми становитиме n^2. Але в такому вигляді хромосома не оптимальна по довжині. При вказаних умовах задачі матриця суміжності є симетричною відносно головної діагоналі, а елементи самої головної діагоналі рівні нулю. Тому в хромосомі присутньо багато зайвої (елементи головної діагоналі) та надлишкової (елементи матриці, що знаходяться під головною діагоналлю) інформації. Ці елементи на рисунку 3.9 позначені сірим кольором. Для скорочення довжини хромосоми вирішено брати до уваги лише елементи матриці, що знаходяться над головною діагоналлю. На рисунку 3.9 ці елементи виділені жирним шрифтом. Виходячи з цього, хромосома виглядатиме наступним чином (рисунок 3.11).

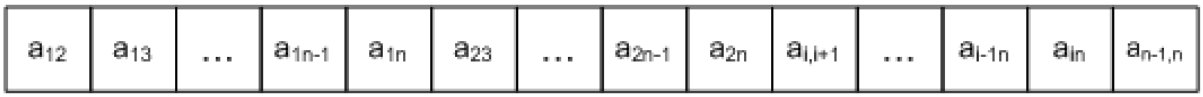


Рис. 3.11 Оптимізована хромосома

Довжина оптимізованої хромосоми становить:

$$Lc = \sum_{i=1}^{n-1}(n-i) = \frac{n(n-1)}{2}. \qquad (3.19)$$

Це значення безперечно менше ніж довжина не оптимізованої хромосоми. З такої хромосоми досить легко відновити матрицю суміжності, перебираючи хромосому зліва направо. Спочатку виділяємо

(n-1) бітів і формуємо першу стрічку, далі – (n-2) бітів та формуємо другу. Так продовжуємо до тих пір, поки не розберемо усю хромосому.

Таким самим чином можна отримати множину лінків *L*. Отримавши матрицю суміжності або множину лінків, визначаються маршрути проходження роутів, інформаційні потоки кожного лінка та ін.

Розробка фітнес-функції. Від побудови хромосоми, тобто кодування розв'язку, залежить і робота фітнес-функції. Фітнес-функція – єдина частина програми, яка знає „що ж відбувається насправді". Усі оператори генетичного алгоритму оперують просто двійковою стрічкою, а фітнес-функція фактично оцінює якість розв'язку, для того, щоб надати особині, представленій хромосомою, певні шанси на виживання і продовження еволюції.

В даній задачі фітнес-функція побудована за алгоритмом, зображеним на рисунку 3.12. Критерії *Pkk* і *Lkk*, що приведені в алгоритмі, обчислюються трохи за іншим принципом.

Критерій *Pkk* обчислюється за формулою (3.20).

$$Pkk = \sum_{p_i \in R^0} p_i \Big/ Pk \quad . \tag{3.20}$$

Таким чином, отримуємо критерій *Pkk*<1, що є зручним для представлення значення фітнес-функції. Чим ближче значення пристосованості до 1, тим „якісніший" розв'язок.

Розрахунок *Lkk* побудовано на тому ж принципі приведення критерію до діапазону [0,1] (3.6).

$$Lkk = 1 - \frac{\sum_{len_i \in L^*} len_i}{\sum_{len_j \in L^0} len_j}. \tag{3.21}$$

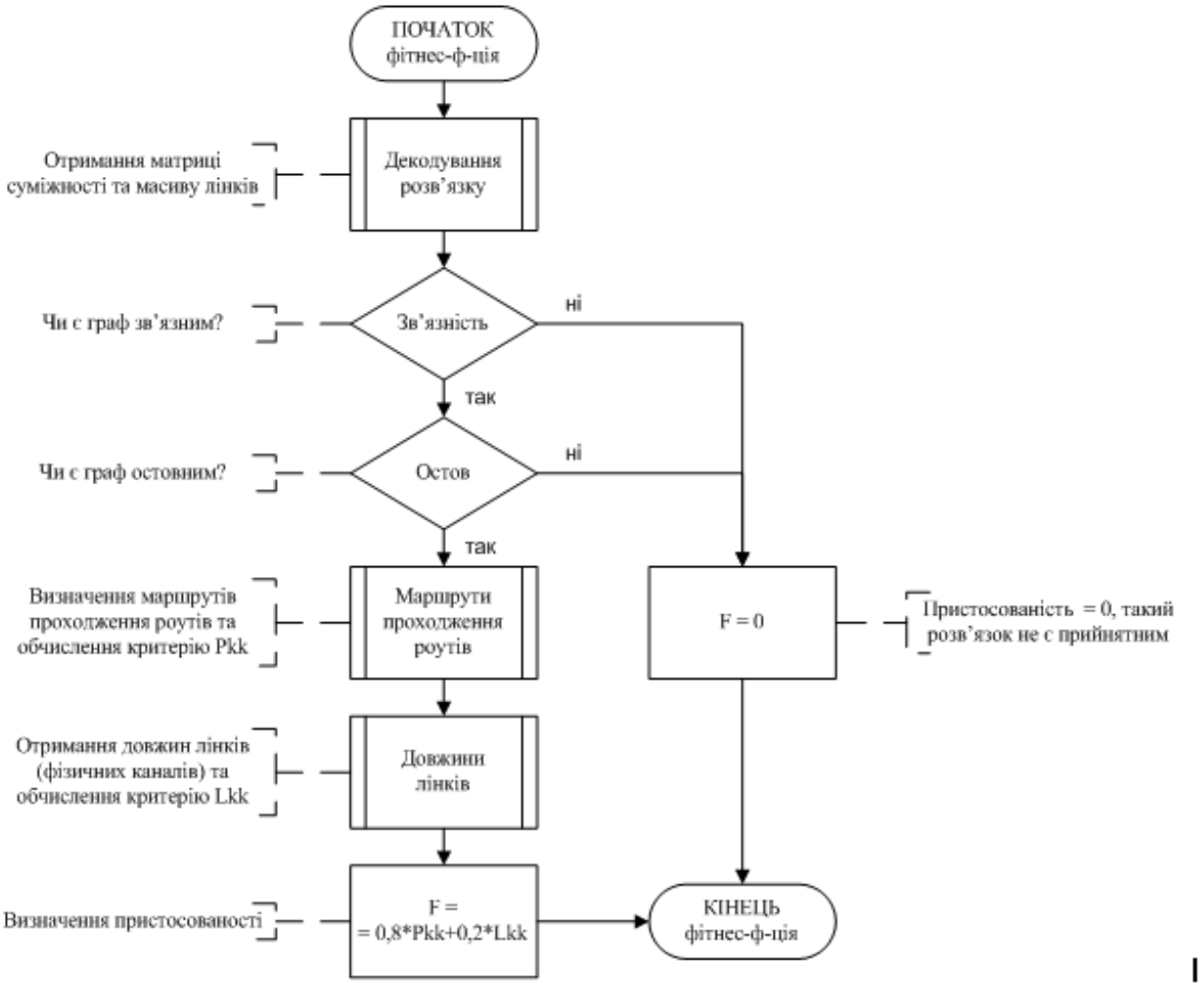


Рис. 3.12 Алгоритм обчислення пристосованості

Остаточне значення фітнес-функції F (3.22) також знаходиться в діапазоні [0,1]. Можна регулювати важливість критеріїв шляхом зміни значень коефіцієнтів при *Pkk* та *Lkk*.

$$F = C_1 \cdot Pkk + C_2 \cdot Lkk\,. \qquad (3.22)$$

Слід детальніше зупинитися на розв'язках, що не задовольняють умові задачі. Пристосованість цих розв'язків дорівнює 0. До таких розв'язків відносяться незв'язні графи та графи, ребра яких утворюють цикли. Незв'язні графи – це такі, в яких є хоча б одна вершина, з якої недосяжні усі інші вершини. Граф, що містить n вершин та (n-1) ребер (тобто не містить циклів), називається деревом або кістяком.

Перевірка графа на зв’язність відбувається шляхом пошуку довжин шляхів між усіма вершинами за допомогою алгоритму Флойда-Уоршела [31]. Для цього матриця суміжності перетворюється на матрицю відстаней, де усі „1” замінюються на відстань між відповідними вершинами, а „0” – якимось відносно великим до відстаней числом (наприклад, 10^9). Далі обчислюються довжини шляхів між усіма вершинами за допомогою простої процедури:

```
for k:=1 to NodeCount do
for i:=1 to NodeCount do
for j:=1 to NodeCount do
D[i,j]:=min(D[i,j],D[i,k]+D[k,j]);
```

Якщо, переглядаючи усю матрицю відстаней D, знайдено хоча б один елемент, рівний 10^9, то шляху між відповідними вершинами немає, а граф є незв’язним.

Перевірка, чи є граф кістяковим, виконується просто: для цього у зв’язного графа має бути рівно (n-1) ребер.

Хоча і недоцільно використовувати генетичний алгоритм на задачах, для яких функція пристосованості складно або довго обчислюється, все ж вирішено застосувати еволюційний підхід з метою вивчення можливостей генетичних алгоритмів при розв’язку подібного класу задач.

Вибір типів операторів генетичного алгоритму. Від вибору тих чи інших типів основних операторів генетичного алгоритму залежить, як швидкість пошуку розв’язку, так і якість розв’язку. З множини варіантів генетичних алгоритмів за основу обрано канонічний генетичний алгоритм (фіксований розмір популяції, фіксована розрядність генів, особини для схрещування вибираються випадковим чином, одно точковий кросовер і одно точкова мутація) з деякими модифікаціями. Всі оператори генетичного алгоритму вирішено писати самостійно з метою глибшого розуміння механізмів роботи базових маніпуляцій над особинами.

Початкова популяція задається шляхом створення визначеного числа

особин із застосуванням генератору випадкових чисел та обчислення пристосованостей особин:

```
SetLength(Population,PopSize);                  //чисельність популяції
for i:=0 to PopSize-1 do                        //для кожної особини
with Population[i] do begin                     //в популяції
Chromosome:=GeneratePopItem(0.25);              //визначити її генотип (0.25 – ймов.1)
Fittness:=ItemFittness(Chromosome);             //та пристосованість
end;
```

Хромосоми особин генеруються з врахуванням ймовірності рівності біту „1”. В загальному випадку ймовірність одиничного біту дорівнює 0,5. Але, враховуючи умови задачі, зокрема те, що граф має бути кістяковим (тобто кількість ребер повинна бути малою), вирішено генерувати хромосоми з ймовірністю одиничного біту не більше 0,25 для можливої появи вже в початковому поколінні особин з ненульовою пристосованістю. Таке рішення сприяє швидшому пошуку розв’язку.

Виходячи з того, що в поколінні може бути багато особин з нульовою пристосованістю, відбір для схрещування відбувається випадковим чином. Кожна особина, незалежно від пристосованості, має однакові шанси на утворення нащадків. Адже навіть „непристосовані” особини можуть при схрещуванні дати „якісний” розв’язок.

Схрещування особин відбувається шляхом одно точкового кросинговеру. Для цього випадковим чином вибирається точка кросинговеру, що поділяє хромосому на дві частини. Батьки обмінюються одними з частин, утворюючи два нащадки.

Після схрещування з деякою ймовірністю може відбутися мутація. Зазвичай, ймовірність мутації знаходиться в межах від 0,01 до 0,1. Ймовірність мутації зростає, якщо батьківські особини схожі між собою. Схожість батьків визначається величиною хемінгової відстані між хромосомами. Якщо хемінгова відстань не перевищує 5% від довжини хромосоми, то ймовірність мутації зростає в декілька разів (в даній реалізації – в 5 разів). Такий крок дозволяє середній пристосованості

популяції зростати швидше і попереджує явище збіжності, коли усі особини в поколінні мають однакові хромосоми.

Операції відбору та схрещування відбуваються (*PopSize* **div** 2) раз, де *PopSize* – розмір популяції. Таким чином, утворюються стільки ж нащадків, скільки і батьків.

Формування наступного покоління відбувається за наступною схемою. Спочатку масиви батьків та нащадків впорядковуються за спаданням значення пристосованості. Далі, до нового покоління відбираються половина батьків та половина нащадків. Отже, до нового покоління потраплять найкращі особини: як батьки, так і нащадки.

Вибір критерію зупинки та налаштування параметрів генетичного алгоритму. Критерій зупинки роботи генетичного алгоритму визначає момент закінчення розвитку популяції. В якості критеріїв зупинки, можливих в програмній реалізації задачі, прийнято наступні:

- досягнення максимальною пристосованістю певного значення, яке задає користувач;

- проходження певної кількості поколінь (значення задає користувач);

- комбінація двох попередніх критеріїв: якщо спрацює хоч один з них, то процес пошуку розв’язку завершується.

На ефективність роботи генетичного алгоритму впливають наступні його параметри, що піддаються налаштуванню:

- розмір популяції (10 – 50);

- ймовірність кросинговеру (0,5 – 0,9);

- ймовірність мутації (0,01 – 0,1).

Робота по настроюванню параметрів покладається на користувача, саме він задає показники роботи алгоритму, залежно від бажання.

Аналіз ефективності генетичного алгоритму програми ОПТІКОС. Деякі зауваження щодо ефективності розробленого генетичного алгоритму для розв’язання задачі про оптимальний розподіл інформаційних потоків.

В ході проведення експериментів щодо перевірки ефективності генетичного алгоритму, використовуваного програмою „ОПТІКОС” [22],

виявлено цікаву закономірність. На швидкість знаходження розв'язку впливають не лише параметри алгоритму, але і спосіб генерації початкового покоління. Якщо бути точнішим, то саме ймовірність одиничного біту. Очевидно, що при застосованій схемі кодування розв'язків, для отримання кістякового дерева одиничних бітів (тих, що рівні „1") має бути менше, ніж нульових. При цьому прийнятний розв'язок буде досягатися швидше. Так як кількість ребер в кістяковому дереві дорівнює (n-1), то рекомендована

Ймовірність одиничного біту обчислюється за формулою:

$$p_1 = \frac{n-1}{Lc} = \frac{n-1}{\frac{n\cdot(n-1)}{2}} = \frac{2}{n}, \text{ де } n \text{ – кількість вершин (вузлів) графу} \qquad (3.23)$$

В якості базової задачі для дослідження залежності швидкості розв'язку від ймовірності одиничного біту та усіх інших дослідів, приведених нижче, вибрано задачу з 10 вузлами та 17 роутами. Граф задачі зображено на рисунку 3.13. Дослід виконувався на комп'ютері Intel Celeron 433 MHz з 192 MB оперативної пам'яті під операційною системою Windows XP Professional.

Так, при використанні рекомендованої ймовірності генетичний алгоритм набагато швидше знаходив ненульовий розв'язок, що значно прискорювало подальшу еволюцію популяції.

Швидкість знаходження розв'язку в залежності від ймовірності одиничного біту виглядає, як показано на рисунках 3.14 та 1.15.

Для оцінки швидкості генетичного алгоритму в залежності від розмірності задачі проведено порівняння часу пошуку розв'язку при різній розмірності задачі.. Особливістю досліду є те, що для кожної розмірності пошук тривав визначену кількість поколінь (10000 поколінь). Такий підхід дозволяє оцінити час обчислення фітнес-функції та виконання генетичних операторів залежно від розмірності задачі, чим, в основному, і визначається швидкість роботи генетичного алгоритму.

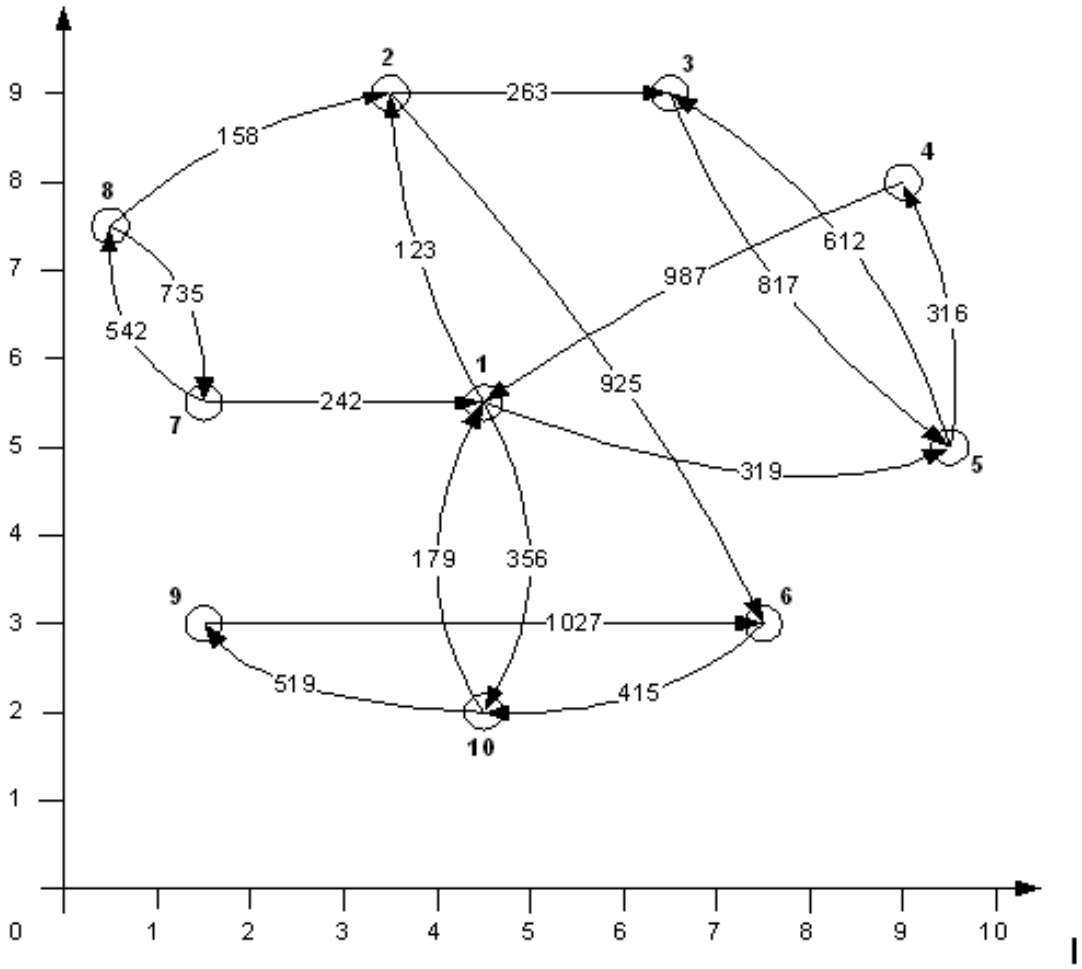


Рис. 3.13 Граф базової моделі

Причому значення ймовірності одиничного біту при генерації початкового покоління підбиралося оптимальним чином за формулою 3.23.

Для розв’язку кожної задачі були прийняті наступні параметри:

- розмір популяції – 20;
- ймовірність кросинговеру – 0,9;
- ймовірність мутації - 0,1.

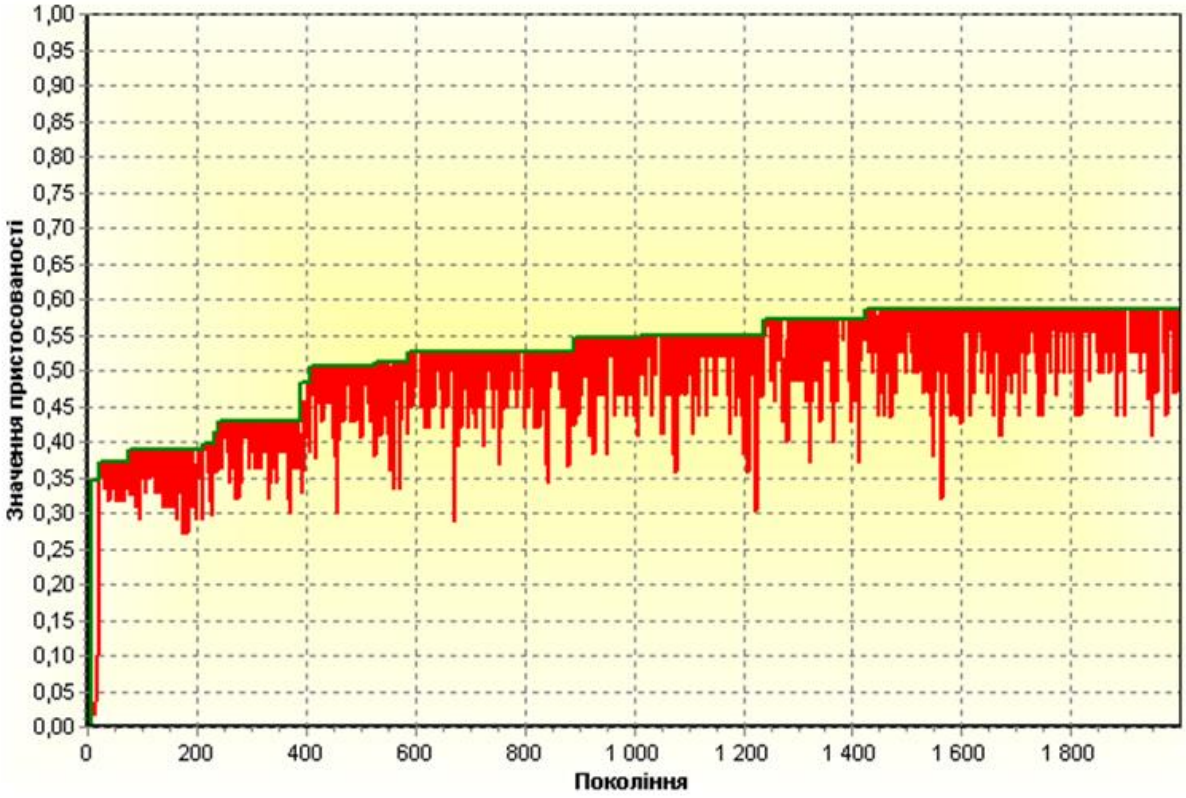


Рис. 3.14 Ріст пристосованості при оптимальному розрахунку ймовірності одиничного біту

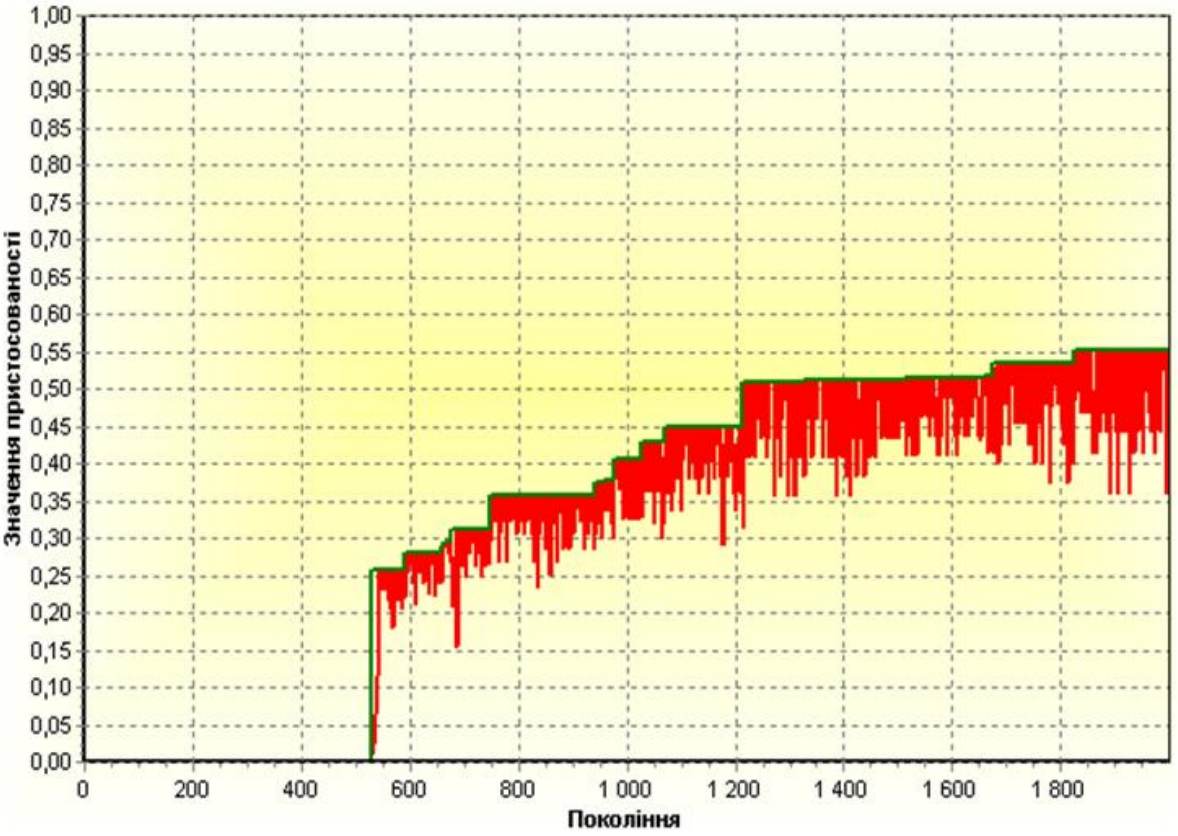


Рис. 3.15 Ріст пристосованості при ймовірності одиничного біту, рівній 0,5

Всього було розглянуто 5 різних задач:

- 6 вузлів та 11 роутів (6N11F);
- 10 вузлів та 17 роутів (10N17F);
- 15 вузлів та 23 роути (15N23F);
- 20 вузлів та 27 роутів (20N27F);
- 25 вузлів та 41 роут (25N41F).

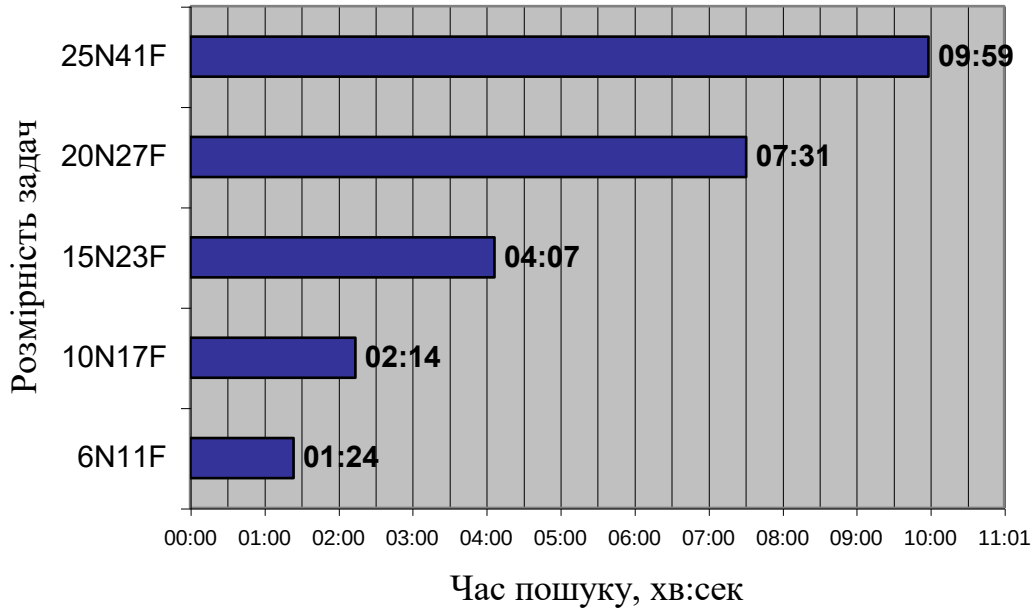


Рис. 3.16 Залежність часу пошуку від розмірності задачі

Як і слід було очікувати, з ростом розмірності задачі зростає і час пошуку. Це цілком закономірно, адже з ростом розмірності ускладнюється розрахунок функції пристосованості, а також зростає розмір хромосоми, що вповільнює роботу генетичних операторів. В усіх запусках без виключення було знайдено хоча б якийсь розв’язок. Це свідчить про те, що навіть при більших розмірах задачі алгоритм дає прийнятний розв’язок за достатньо короткий час порівняно з алгоритмами повного перебору.

Великий вплив на швидкість отримання розв’язку та його якість має також і ймовірність мутації. Як відомо, ймовірність мутації визначає різноманітність популяції. Чим вища ймовірність, тим різноманітніша популяція. Розвиток не зосереджується навколо якогось розв’язку, а

продовжує пошук нових розв’язків у інших точках ландшафту функції.

З метою вивчення впливу ймовірності мутації на швидкість еволюції розв’язку проведено наступний дослід. Оцінювалось значення функції пристосованості, досягнуте на протязі 3000 поколінь при різних ймовірностях мутації. Так як генетичні алгоритми несуть деякий елемент випадковості, то бралося до уваги середньоарифметичне значення досягнутої пристосованості на основі 5 запусків. Слід також зауважити, що в досліджуваній програмі використовувалося явище п’ятикратного збільшення ймовірності мутації нащадків при спорідненості особин батьківської пари.

Вплив ймовірності мутації на швидкість росту пристосованості проілюстровано на рис. 3.17.

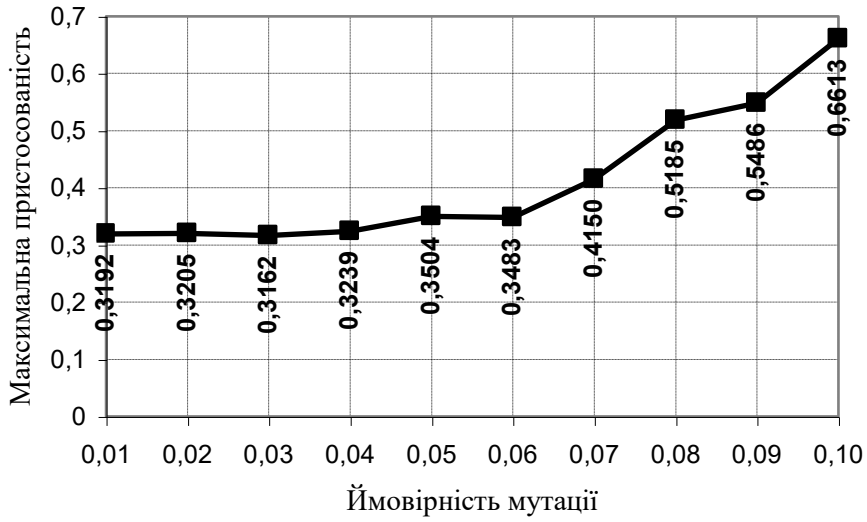


Рис. 3.17 Залежність часу пошуку від розмірності задачі

Майже однакову пристосованість для ймовірності мутації від 0,01 до 0,06 можна пояснити тим, що максимальна пристосованість, яка була отримана при генерації початкового покоління (використовувався оптимальний показник ймовірності одиничного біту згідно формули 3.23) або в перші покоління розвитку, зберегла своє значення до кінця процесу пошуку розв’язку.

3.4.2 Метод пошуку раціональних «зіркоподібних» структур інформаційних систем з використанням кодування Прюфера

В попередньому п. 3.4.1 запропоновано розширення постановки задачі, описаної в п. 3.2 і модифікований класичний генетичний алгоритм її рішення. Структури, які формуються при цьому, мають "зіркоподібний" вигляд, але система кодування, що використовується в алгоритмі, охоплює всі види структур. Приклад домінуючих "зіркоподібних" структур - структура ІКОС для ІПС СС, рис. 3.7). Обмеження типів структур, а також введення в генетичному алгоритмі чисел Прюфера для їх кодування, дозволяє зменшити час пошуку раціональних рішень. Крім того, цікавим для дослідження є використання можливостей пакету MATLAB, його бібліотеки GATool, для реалізації запропонованих алгоритмів. Це дозволить використовувати отримані результати не лише в процесі проектування АСК, але й в навчальному процесі при вивченні методів рішення інженерних задач в MATLAB.

Розглянемо алгоритми реалізації розв'язку задачі.

Кодування та декодування розв'язків задачі оптимізації. Для роботи генетичного алгоритму необхідно визначитися з типом особин популяції, тобто з представленням розв'язків завдання оптимізації.

При виборі способу кодування кістякових дерев необхідно враховувати, який вигляд приблизно може мати оптимальна структура ІКОС. Якщо не враховувати обмеження на об'єм трафіку в каналі, то оптимальною структурою інформаційної системи буде структура «зірка» (рис. 3.18), при якій одна підсистема знаходиться в центрі «зірки», інші - «променями» з'єднуються з центром.

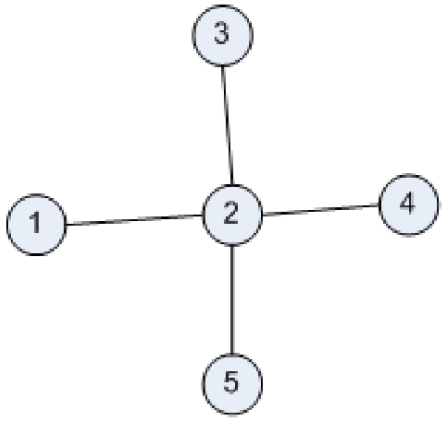


Рис. 3.18 Структура «зірка»

Оптимальність такої структури в рамках даного методу забезпечується мінімальним приростом сумарного інформаційного потоку через те, що дублювання інформаційних потоків у системі – мінімальне. Якщо враховувати обмеження на об'єми трафіку, то, ймовірніше всього, оптимальною структурою буде деяка зіркоподібна структура, так званий розвиток «зірки» (рис.3.19).

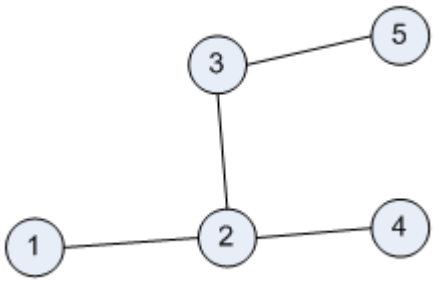


Рис. 3.19 "Зіркоподібна" структура

Для кодування кістякових дерев обрано код Прюфера, який представляє собою спосіб однозначного кодування дерева за допомогою послідовності чисел. Згідно теореми Келі [25], на N вершинах, що пронумеровані натуральними числами від 1 до N, існує рівно N^{N-2} різних кістякових дерев. Прюфер довів, що існує однозначне перетворення між кістяковими деревами з N вершинами і векторами довжиною $(N-2)$, в яких кожен елемент є натуральне число від 1 до N [200].

Алгоритм кодування структури кістякового дерева наведено на рис. 3.20. В масиві code в кінці роботи алгоритму знаходиться код Прюфера відповідного дерева.

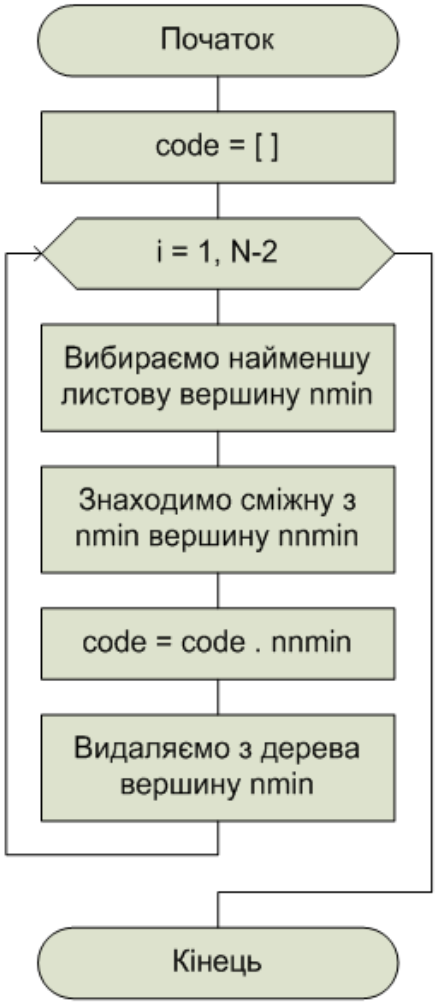


Рис. 3.20 Алгоритм кодування дерев кодом Прюфера

Розглянемо приклад кодування (рис. 3.21).

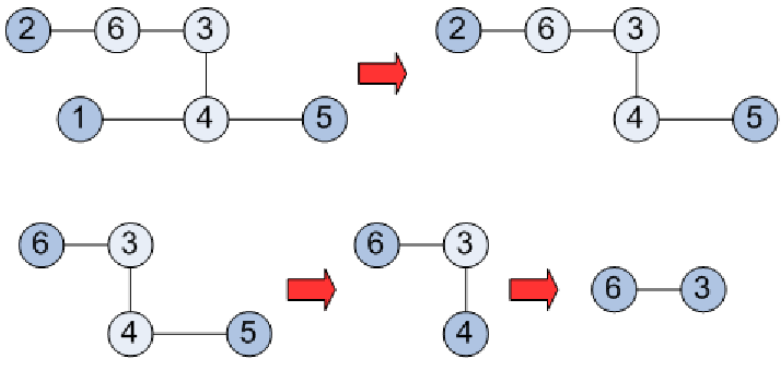


Рис. 3.21 Процес кодування дерева

Вершина 1 є найменшою листовою вершиною графу. Вершина 4 є суміжною з вершиною 1. Тобто, цифра 4 буде першою в коді Прюфера. Видаляємо вершину 1 і ребро (1,4). Повторюємо процес, доки не залишиться ребро (3,6). Отриманий код Прюфера для прикладу з рис. 3.21 – [4 6 4 3].

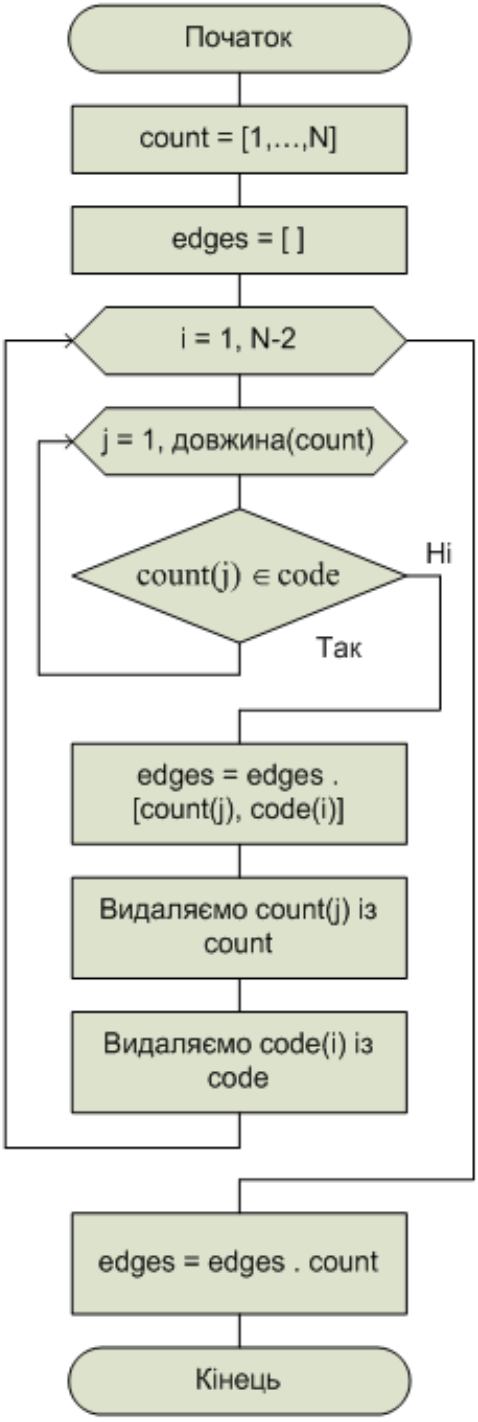


Рис. 3.22 Алгоритм декодування дерев числами Прюфера

Алгоритм декодування дерева з коду Прюфера наведена на рис. 3.23.

Розглянемо приклад декодування кода Прюфера [3 2 2 1] (див. рис.3.23).

Для коду Прюфера P = [3 2 2 1] набір вершин, що відсутні у коді – $\overline{P}$ = [4 5 6]. Вершина 4 є найменшою з $\overline{P}$, а вершина 3 є першою у P. Додаємо ребро (3,4) до дерева, видаляємо вершину 3 з P і вершину 4 з $\overline{P}$. Оскільки вершина 3 тепер відсутня у P, додаємо її в $\overline{P}$. На наступному кроці маємо P = [2 2 1], $\overline{P}$ = [3 5 6]. Повторюємо зазначені вище дії, поки в P не залишиться елементів, а $\overline{P}$ = [1 6]. Додаємо до дерева останнє ребро (1,6). На цьому процес декодування завершується.

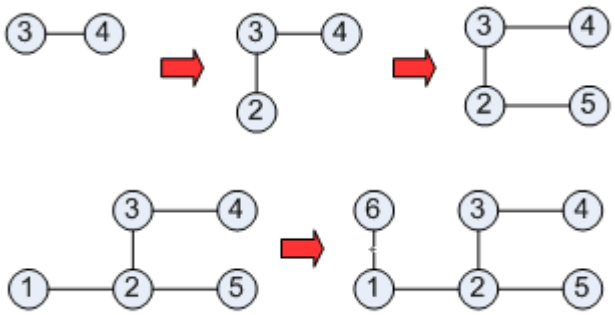


Рис. 3.23 Процес декодування дерева

Код Прюфера широко використовується дослідниками при розв'язку MST-завдань [185]. Даний код демонструє свою ефективність при пошуці саме зіркоподібних кістякових дерев. «Зірка» з N вершин в коді Прюфера представляє собою послідовність довжиною $(N-2)$, в якій всі елементи рівні. Наприклад, послідовність [3 3 3] описує «зірку» із 5 вершин з центральною вершиною 3. Таку властивість коду можна використати для прискорення пошуку оптимальної структури при формуванні першого покоління алгоритму.

Через те, що тип особин генетичного алгоритму при використанні коду Прюфера – цілочисельні рядки, в MATLAB необхідно самостійно реалізувати процедури ініціалізації популяції, оператор кросинговеру та мутації.

Ініціалізація популяції. Процес ініціалізації популяції проходить на початку роботи генетичного алгоритму. Реалізація даної процедури може значно прискорити пошук оптимальної структури, чим підвищується ефективність усього алгоритму.

Зважаючи на те, що оптимальним розв'язком задачі оптимізації ІС найчастіше є «зірка» чи її розвиток, є сенс в ініціалізації частини початкового покоління усіх можливих «зірок» для заданої кількості вузлів. Інша частина популяції є повністю випадковою для внесення більшої генетичної різноманітності. Таким чином, якщо за умовами задачі, на вузол не накладають обмеження на трафік, а також розмір популяції є достатнім для ініціалізації усіх можливих «зірок», оптимальний розв'язок буде знайдений вже на першій ітерації генетичного алгоритму.

Алгоритм ініціалізації популяції наведений на рис. 3.24.

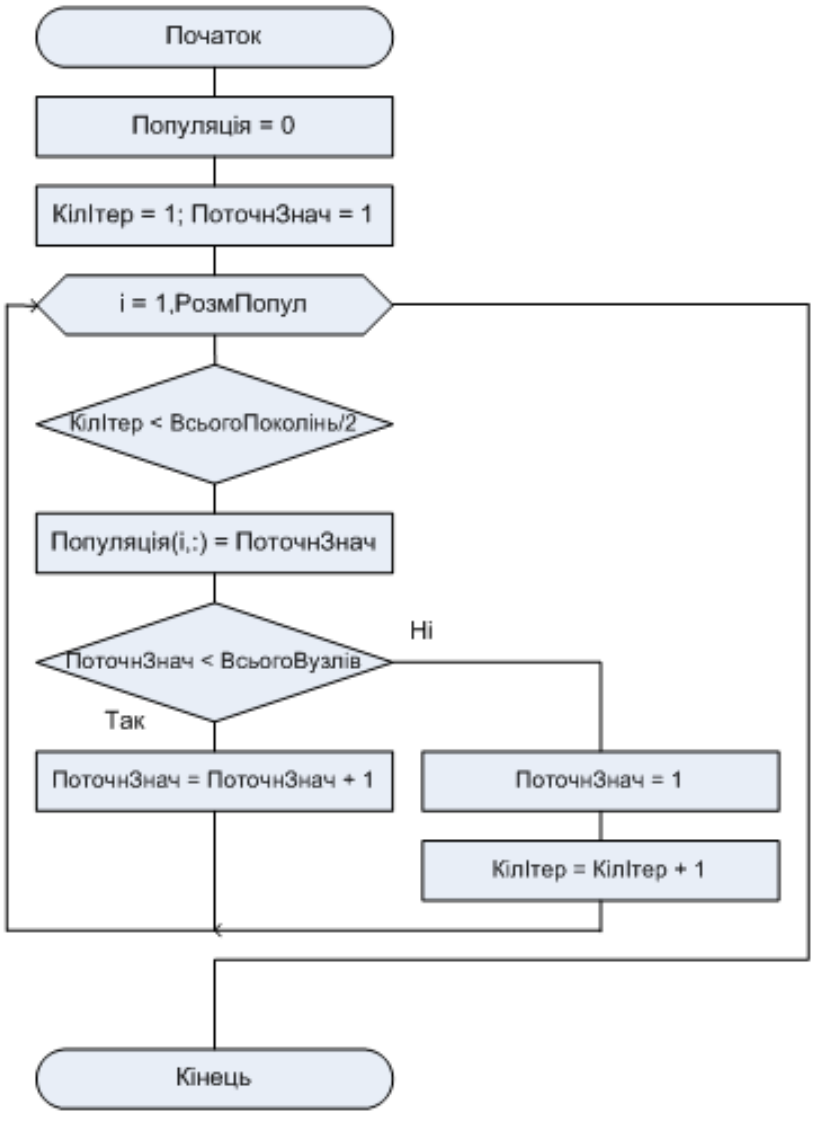


Рис. 3.24 Алгоритм ініціалізації популяції

Оператор кросинговеру. Через специфічний тип особин генетичного алгоритму був розроблений спеціальний оператор кросинговеру. В [200] для розв'язку задач MST (Minimum Spanning Tree) рекомендується використовувати двоточковий кросинговер, який э ефективніше одноточкового. Особливістю запропонованої реалізації кросинговеру є утворення лише одного нащадка, при чому порядок обміну батьків своїми фрагментами є випадковим.

Алгоритм реалізації кросинговеру наведений на рис. 3.25.

Оператор мутації. Для більш ефективної боротьби з передчасної збіжністю, а також для внесення більшої різноманітності в популяцію використовуємо двоточкову мутацію. За основу взята стандартна двоточкова мутація для бітових особин, що пропонується утилітою GATool в пакеті MATLAB, модифікована для цілочисельних рядків. До того ж, якщо протягом п'ятьох поколінь алгоритм не покращує розв'язок, випадкова особина в популяції мутує повністю.

Алгоритм оператору двоточкової мутації наведений на рис. 3.26.

Функція пристосованості. Розробка функції пристосованості є невід'ємною частиною проектування генетичного алгоритму. Функція пристосованості має давати об'єктивну оцінку ефективності кожної особини, враховуючи всі обмеження задачі.

У випадку оптимізації структури ІКОС функція пристосованості має обчислювати приріст сумарного інформаційного потоку в структурі, що закодована в особині популяції, в порівняні з повнозв'язною структурою системи. Функція має декодувати код Прюфера (тобто особину популяції), розраховувати маршрути для всіх інформаційних потоків в системі, на основі яких розраховується приріст сумарного інформаційного потоку, а також враховувати, чи виконуються обмеження на трафік в підсистемах.

Значення функції складається із значення приросту сумарного інформаційного потоку, а також з сумарного перевищення встановлених обмежень на трафік всіх вузлів системи. Таким чином, чим менше значення функції пристосованості, тим кращий розв'язок пропонує особина популяції.

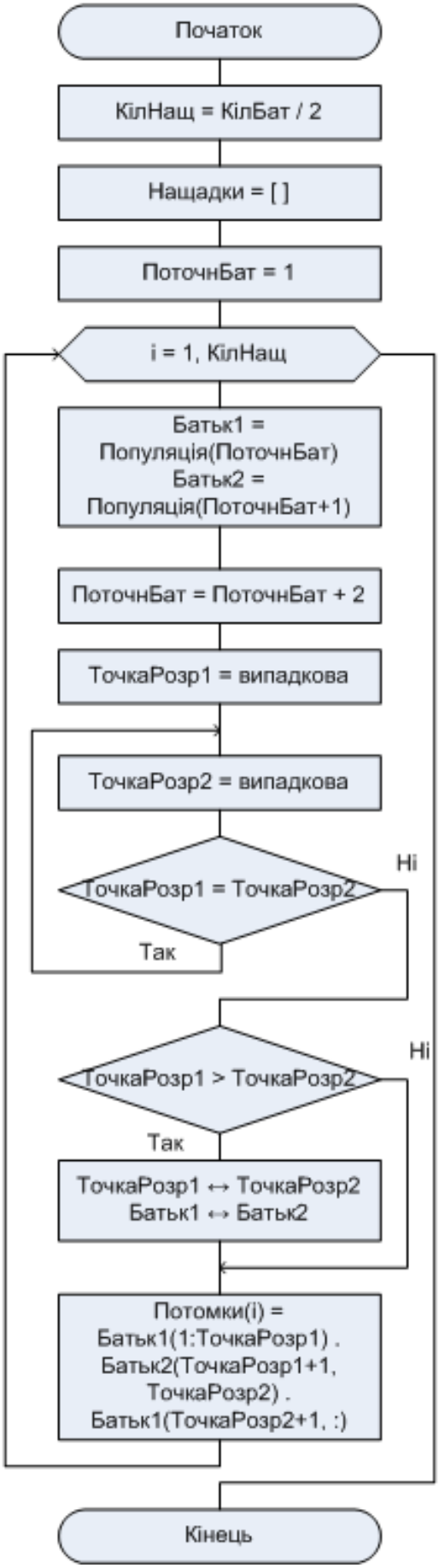


Рис. 3.25 Алгоритм оператору кросинговеру

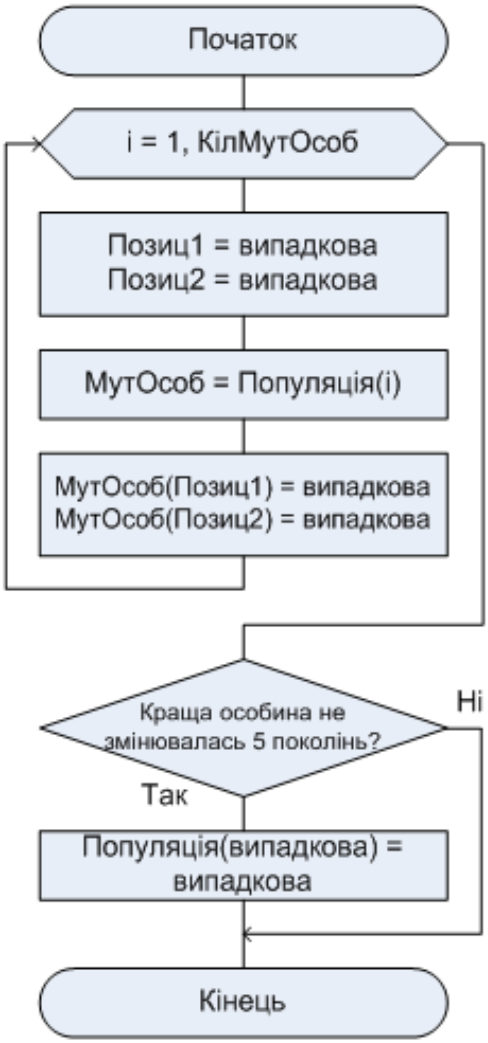


Рис. 3.26 Алгоритм оператору мутації

Алгоритм розрахунку функції пристосованості наведений на рис. 3.27.

Зауваження щодо програмної реалізації розв'язку задачі оптимізації. На основі алгоритмів, описаних вище, на мові програмування MATLAB була реалізована програма GAOSIS [20], що виконує оптимізацію структури інформаційної системи. Вхідними даним для неї є матриця потоків даних та матриця обмежень трафіку. Результатом роботи програми є набір ребер графу. На рисунку 3.28 представлена візуалізація підсумкової структури системи.

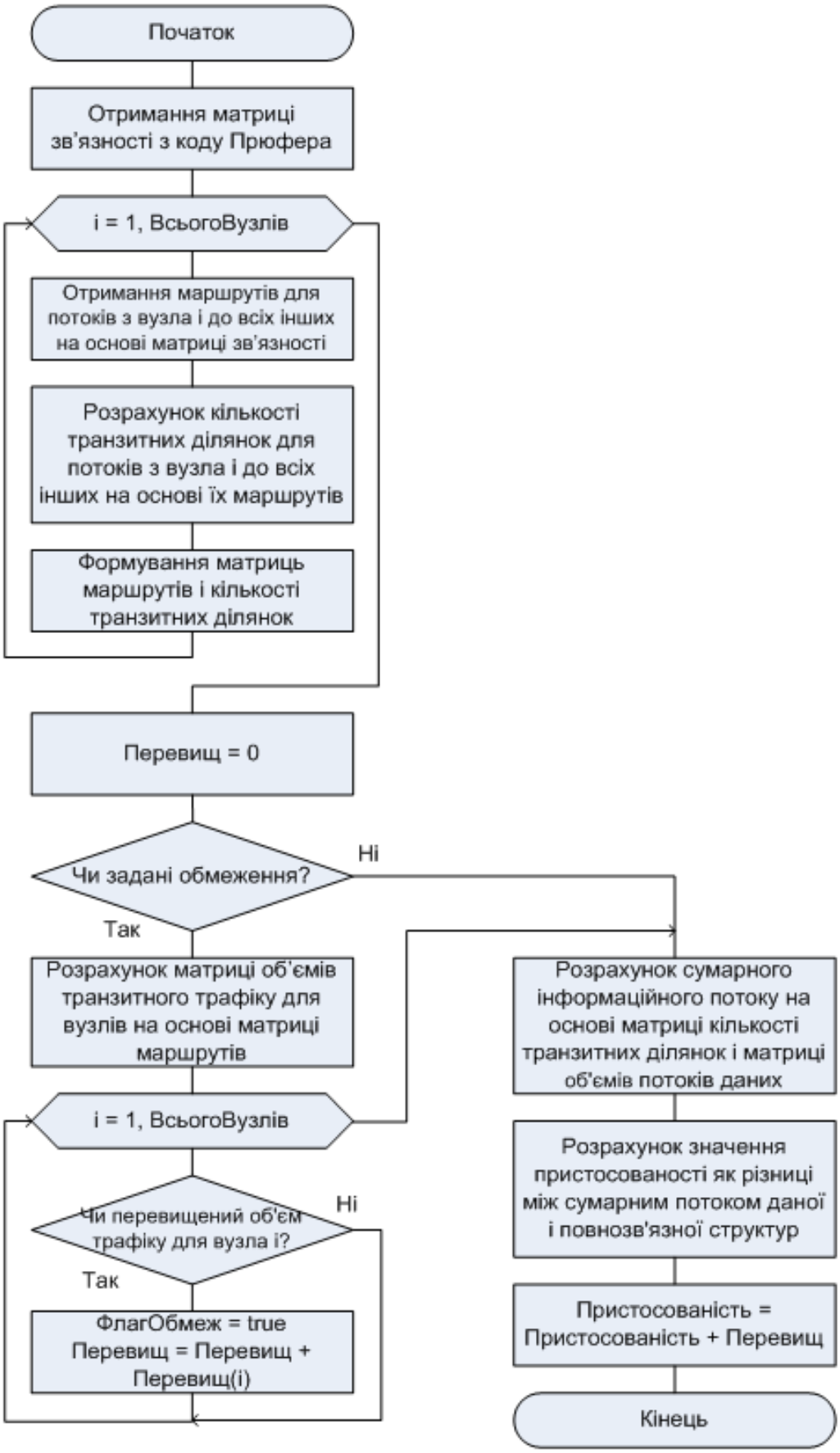


Рис. 3.27 Алгоритм функції пристосованості

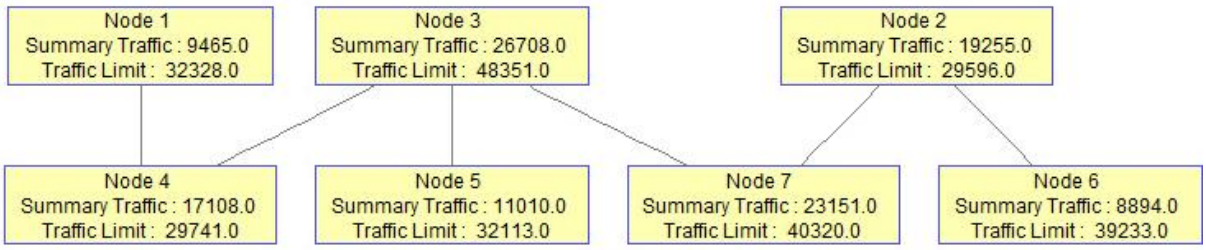


Рис 3.28 – Візуалізація розв'язку задачі оптимізації структури

3.5 Ресурсоощадні методи вибору технічних структур децентралізованих інформаційно-керуючих систем залізничних сортувальних станцій

3.5.1 Ресурсоощадний метод дослідження умов доцільності децентралізації функцій керування

Розвиток систем автоматизації сортувальних гірок в Україні і за кордоном походило від створення перших централізованих комп'ютерних систем на базі міні-ЕОМ (СМ-2) до побудови функціонально розподілених систем на основі промислових мікро-ЕОМ і мікроконтролерів (мікро-ДАТ, СМ-1800/1810, мікроконтролери Advantech) [69, 158]. З переходом на децентралізовані (ієрархічні) системи при різноманітності варіантів розподілених структур невирішеним залишилося питання обґрунтованості переходу до децентралізованого керування. Не дивлячись на численні спроби знайти відповідь на дане питання [24, 47, 108, 150] воно залишається актуальним в умовах розвитку технічних засобів автоматизації сортувальних гірок.

Метою даної роботи є дослідження умов доцільності створення децентралізованих систем. Це завдання є актуальним і для крупнішої

інтегрованої автоматизованої системи керування вантажними перевезеннями УКРЗАЛІЗНИЦІ - АСК ВП УЗ-Є [69].

Під час розгляду питань оптимізації структури ієрархічної системи ми будемо вважати, що сортувальна гірка, як технологічний об'єкт керування сортувальної станції, має локальні пристрої автоматики.

Централізована система керування, оптимізуючи в цілому роботу сортувальної гірки, корегує роботу місцевих пристроїв автоматики, формуючи уставки для регуляторів (наприклад, гальмівних позицій, стрілкових переводів). За таким принципом була побудована система АСК РСГ для станції Ясинувата Донецької залізниці [69].

Зі зростанням потужності керуючих обчислювальних машин (КеОМ) виникає інтерес розглянути доцільність передачі функцій місцевих пристроїв автоматики керуючим ЕОМ більш високого рангу, які реалізують алгоритм оптимізації режиму роботи системи [150].

Якщо раніше можливості систем гіркової автоматики були обмеженими. з погляду реалізації складних алгоритмів керування, а КеОМ не відрізнялися великою швидкодією, то зараз є технічні можливості нарощувати потужності як мікроконтролерів, так і керуючих ЕОМ. Тому передача функцій місцевих регуляторів на КеОМ істотно не вплине на її потужність (обсяг пам'яті, швидкодію і т. і.).

Однак така централізація управління суттєво підвищує відповідальність КеОМ. Очевидно, що вихід з ладу КеОМ порушує управління цілою сортувальною гіркою, що призводить майже завжди до великих економічних втрат. Тому багато авторів вважають [126, 128, 144], що визначальним у вирішенні цього питання є надійність КеОМ.

Сортувальна гірка О складається з n керованих об'єктів, стан кожного з котрих O_i можна охарактеризувати двома параметрами: y_i - регульований параметр; x_i – керуюча дія (z_j - контрольований, f_g - неконтрольований параметри, див. рис. 3.29).

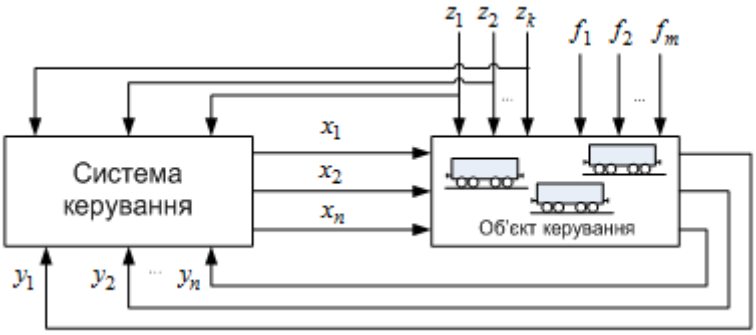


Рис. 3.29 Сортувальна гірка як об'єкт керування

Таким чином, стан системи в кожен момент часу, може бути охарактеризований векторами: стану регульованого параметра $Y = \{y_1, ..., y_n\}$; положення регулюючого органу $X = \{x_1, ..., x_n\}$ і вектором-уставкою $Y' = \{y'_1, ..., y'_n\}$. Вектор-уставка Y' являє собою завдання на підтримку регульованого параметра і в загальному випадку може відрізнятися від його істинного значення – вектора *Y*.

Якщо система не має локальних регуляторів, то КеОМ безпосередньо формую вектор *X* і видає його на виконавчі органи. В даному випадку ми маємо централізовану систему (див. рис. 3.30). У разі наявності місцевих регуляторів (МР), що реалізують функції локального керування, функції КеОМ спрощуються до формування лише уставок Y' для МР. В цьому разі говорять про ієрархічну структуру системи керування (див. рис. 3.31).

Нехай система складається з КеОМ і *n* локальних підсистем.

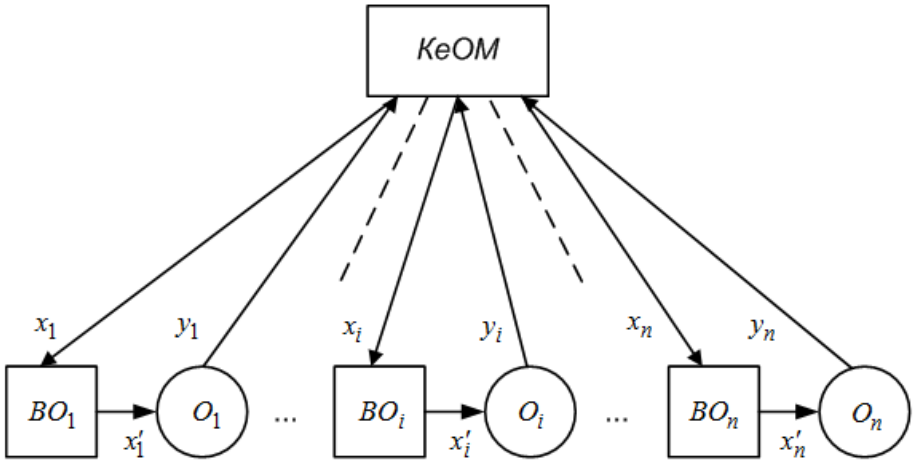


Рис. 3.30 Структура централізованої системи керування гіркою із застосуванням КеОМ без автономних підсистем управління.

На рис. 3.30 і 3.31 прийнятні наступні позначення: КеОМ – керуюча обчислювальна машина; МР – місцевий регулятор; O_i – *i*-а підсистема; ВО – виконавчий орган.

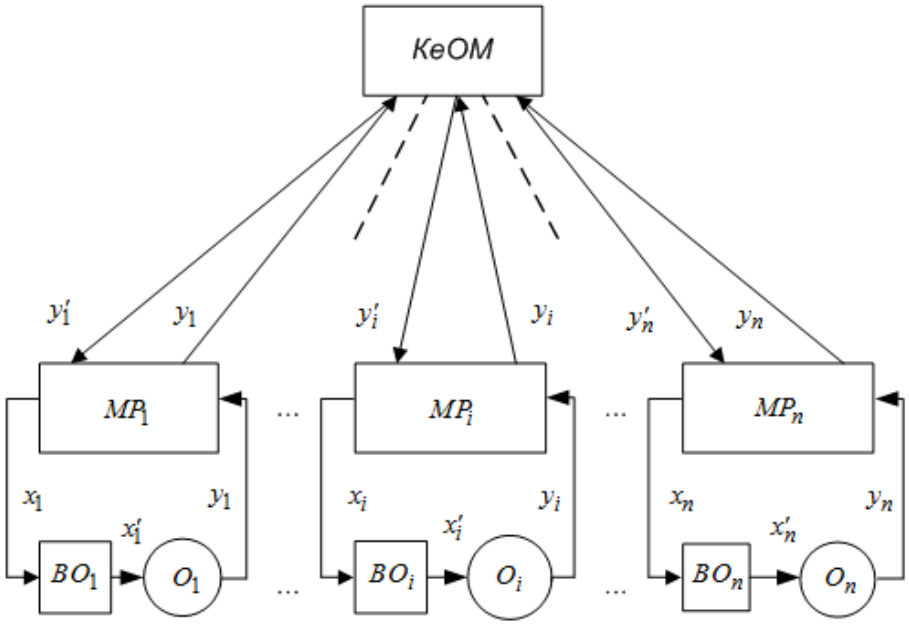


Рис. 3.31 Структура ієрархічної системи із застосуванням КеОМ і місцевих підсистем керування

Реалізація алгоритму управління зводиться до мінімізації (максимізації) функції виду

$$B = \varphi(Y, X) \tag{3.24}$$

(наприклад, мінімізації відхилення швидкості виходу з гальмівної позиції від заданої за алгоритмом керування).

Оптимізація передбачає пошук такого оптимального значення вектора керування *U**, щоб функціонал управління (3.24) при даному *X* приймав оптимальне значення

$$B^* = \min(\max)\varphi(Y, X) \tag{3.25}$$

або

$$B^* = \varphi\left(Y, X^*\right). \tag{3.26}$$

Оцінку варіантів структур будемо проводити за критерієм повних витрат ресурсів. Для спрощення розрахунків приймаємо, що всі n контурів керування однакові, витрати на експлуатацію систем також однакові і ненадійністю виконавчих органів можна знехтувати.

Керування системою будемо представляти в режимі розподілу часу наступним чином. За схемою на рис. 3.30 (централізована система керування) КеОМ через інтервали часу $T_{\text{ц}}(T_{\text{ц}}, 2T_{\text{ц}}, ..., kT_{\text{ц}})$ циклічно отримує інформацію про стан керованого об'єкта у вигляді векторів Y. В результаті переробки інформації за алгоритмом (3.25) КеОМ здійснює розрахунок нової уставки Y' і видачу команди керування на виконавчі органи – вектор положення X.

За схемою на рис. 3.31 (ієрархічна система управління) КеОМ також отримує інформацію у вигляді векторів Y. Однак у цьому випадку КеОМ видає тільки вектор-уставку Y'. Розрахунок вектора X здійснюють місцеві регулятори *MP;* КеОМ тільки координує їх роботу, корегуючи уставку-вектор Y'.

Подивимося, що відбудеться у випадку відмови апаратури в k-й інтервал часу.

При виході з ладу КеОМ в централізованій структурі можливі наступні ситуації:

1) негайно припиняється робота сортувальної гірки. Втрати p_0 визначаються простоями об'єкту і можуть бути оцінені повною втратою ефективності за час усунення несправності (відновлення системи) $T_{\text{відн}}$:

$$p_0 = B^* T_{\text{відн}}.$$

2) Якийсь час після відмови КеОМ система продовжує функціонувати нормально і потім відбувається порушення її роботи. В цьому випадку поведінка об'єкту може змінюватися залежно від того, в якому положенні опиняться виконавські органи. Вони можуть: а) залишатися в тому положенні, в якому вони були у момент аварії t_0, тобто $X = X_0$;

б) переводиться в аварійний стан, що характеризується значеннями $X = X_{\text{ав}}$.

У другій ситуації збиток p_1 складатиметься з двох частин (див. рис. 3.32), $y_i = v_i$ - швидкість виходу з i-ї гальмівної позиції, $v_{i\min} < v_i(t) < v_{i\max}$). На ділянці t_1 він пропорційний різниці між оптимальним і дійсним значеннями ефективності:

$$p_{11} = (B^* - B_1)t_1. \qquad (3.27)$$

На ділянці t_2 відбувається вихід параметра $v_i(t)$ за допустимі межі $v_i(t) > v_{i\max}$ і місцевий захист відключає об'єкт, тобто втрати на цій ділянці p_{12} визначаються повною втратою ефективності:

$$p_{12} = B^* t_2.$$

Повні втрати для другої ситуації

$$p_1 = p_{11} + p_{12} = (B^* - B_1)t_1 + B^* t_2 ,$$

де

$$T_{\text{відн}} = t_1 + t_2; \lim_{t_1 \to \infty} p_1 = B^* T_{\text{відн}} .$$

B_1 - ефективність при не оптимальному керуванні в централізованій системі (ручне керування об'єктами сортувальної гірки).

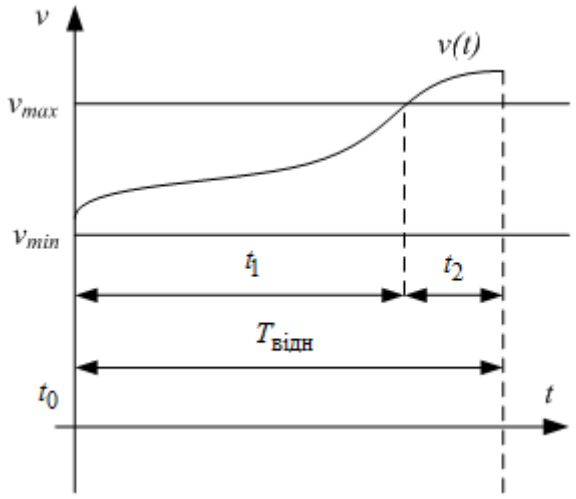


Рис. 3.32 Приклад поведінки регульованого параметра *v(t)* в межах ($v_{мін}$, $v_{макс}$).

У ієрархічній системі при відмові КеОМ управління втрачається не повністю, знижується лише його ефективність. Хай в результаті відмови вона приймає значення B_2. В цьому випадку збиток від втрати оптимального управління

$$p_2 = (B_2^* - B_2)T_{\text{відн}}. \qquad (3.28)$$

Цей збиток, як правило, значно менше збитку, що отримується при виході з ладу КеОМ в централізованій системі. Зменшення збитку в системі з ієрархічною структурою досягається за рахунок збільшення її вартості із-за наявності локальних регуляторів. Крім того, ефект зменшення втрат знижується за рахунок додаткових втрат, що викликаються ненадійністю локальних регуляторів. Приймемо для подальшого аналізу наступні

Припущення:

B^* - втрати в одиницю часу, коли гірка не працює;

B_1 - втрати в одиницю часу, коли в централізованій системі не працює КеОМ (неоптимальне керування);

B_2 - втрати в одиницю часу, коли в ієрархічній системі не працює КеОМ (неоптимальне керування);

b_i^* - втрати в одиницю часу, коли не працює *i*-ий контур керування,

b_i - втрати в одиницю часу при неоптимальному значенні ефективності *i*-го контура.

$$B^* > B_1 > B_2 \quad b_i^* > b_i.$$

Тоді повні витрати для централізованої P_1 і ієрархічної P_2 систем можна оцінити наступним чином:

$$\mathrm{P}_1 = \left(B^* - B_1\right)\lambda_0 t_1 + B^*\lambda_0 t_2 + C_0^{\text{ц}}, \qquad (3.29)$$

$$\mathrm{P}_2 = (B^* - B_2)\lambda_0 T_{\text{відн}} + \sum_{i=1}^{n}\left[(b_i^* - b_i)\lambda_i t_{i1} + b_i^* \lambda_i t_{i2} \right] + C_0^{\text{дц}} + \sum_{i=1}^{n} c_i \text{ , (3.30)}$$

де λ_0 - інтенсивність відмов КеОМ в ієрархічній і централізованій системах; λ_i - інтенсивність відмов і-го локального регулятора; $C_0^{\text{ц}}, C_0^{\text{дц}}, c_i$ - вартість КеОМ відповідно в централізованій системі, в децентралізованій (ієрархічній) системі і вартість мікроконтролерної системи і-го локального регулятора.

Будемо вважати, що

$$C_0^{\text{ц}} > C_0^{\text{дц}} \square \ c_i .$$

Для спрощення подальшого аналізу приймаємо, що система однорідна за своїм складом, тобто

$$b_i = b_n;\ \lambda_i = \lambda_n;\ c_i = c_n;\ i = 1,2,...,n . \qquad (3.31)$$

Для *i*-го контура керування з урахуванням можливого виходу з ладу регулятора по аналогії з централізованою системою управління (див. рис. 3.32) можна написати

$$t_{i2} = T_{\text{відн}\,i} - t_{i1} ,$$

де t_{i2} - час, протягом якого *i*-й параметр знаходиться за допустимими значеннями; $T_{\text{відн}i}$ - час ремонту *i*-го регулятора; t_{i1} - час, протягом якого *i*-й параметр не виходить за допустимі межі.

В першому наближенні приймемо, що

$$\begin{aligned} &t_{i1} = t_{n1} = t_1,\ i = 1,2,...,n; \\ &T_{\text{відн}}^{\text{ц}}(\text{КеОМ}) = T_{\text{відн}}^{\text{дц}}(\text{КеОМ}) = T_{\text{відн}}(c_n). \end{aligned} \qquad (3.32)$$

Для подальшого дослідження введемо коефіцієнт

$$t_1 = rT_{\text{відн}},\ t_2 = (1-r)T_{\text{відн}} .$$

Тоді з (3.29) і (3.30) маємо

$$\mathrm{P}_1 = \left(B^* - B_1\right)\lambda_0 r T_{\text{відн}} + B^*\lambda_0(1-r)T_{\text{відн}} + C_0^{\text{ц}}, \tag{3.33}$$

$$\begin{aligned}\mathrm{P}_2 = {} & (B^* - B_2)\lambda_0 T_{\text{відн}} + (B^* - B_1)\lambda_n r T_{\text{відн}} + B^*\lambda_n(1-r)T_{\text{відн}} + \\ & + C_0^{\text{дц}} + nc_n.\end{aligned} \tag{3.34}$$

Перейдемо до інтегрального показника надійності - коефіцієнта готовності

$$K_{\text{г}}^{\text{ц}}(\text{КеОМ}) = K_{\text{г}}^{\text{дц}}(\text{КеОМ}) = K_0 = \frac{T_{\text{нв}}}{T_{\text{нв}} + T_{\text{відн}}} = \frac{1/\lambda_0}{1/\lambda_0 + T_{\text{відн}}}, \tag{3.35}$$

$$K_{\text{г}}(c_n) = K_n = \frac{T_{\text{нв}(n)}}{T_{\text{нв}(n)} + T_{\text{відн}}} = \frac{1/\lambda_n}{1/\lambda_n + T_{\text{відн}}}, \tag{3.36}$$

де $T_{\text{нв}}$ - середній час напрацювання на відмову.

Звідки

$$\lambda_0 = \frac{1-K_0}{K_0 T_{\text{відн}}};\ \ \lambda_n = \frac{1-K_n}{K_n T_{\text{відн}}}. \tag{3.37}$$

Надалі введемо коефіцієнт α:

$$K_n = \alpha K_0. \tag{3.38}$$

Підставимо (3.37) і (3.38) в (3.33), (3.34), нормалізуємо повні витрати і введемо позначення

$$\delta_1 = \frac{B^* - B_1}{B^*};\ \ \delta_2 = \frac{B^* - B_2}{B^*}.$$

В результаті отримаємо

$$\tilde{\mathrm{P}}_1 = \frac{\mathrm{P}_1}{B^*} = \frac{(1-K_0)(1+r(\delta_1 - 1))}{K_0} + \frac{C_0^{\text{ц}}}{B^*}, \tag{3.39}$$

$$\tilde{P}_2 = \frac{P_2}{B^*} = \frac{\delta_2(1-K_0)}{K_0} + \frac{(1-\alpha K_0)(1+r(\delta_1-1))}{\alpha K_0} + \frac{C_0^{дц} + nc_n}{B^*}, \qquad (3.40)$$

На рисунку 3.33 побудовані графіки $\tilde{P}_1 = f_1(K_0)$ и $\tilde{P}_2 = f_2(K_0)$ при наступних припущеннях $r = 0{,}9$ і $\alpha = 1{,}1$.

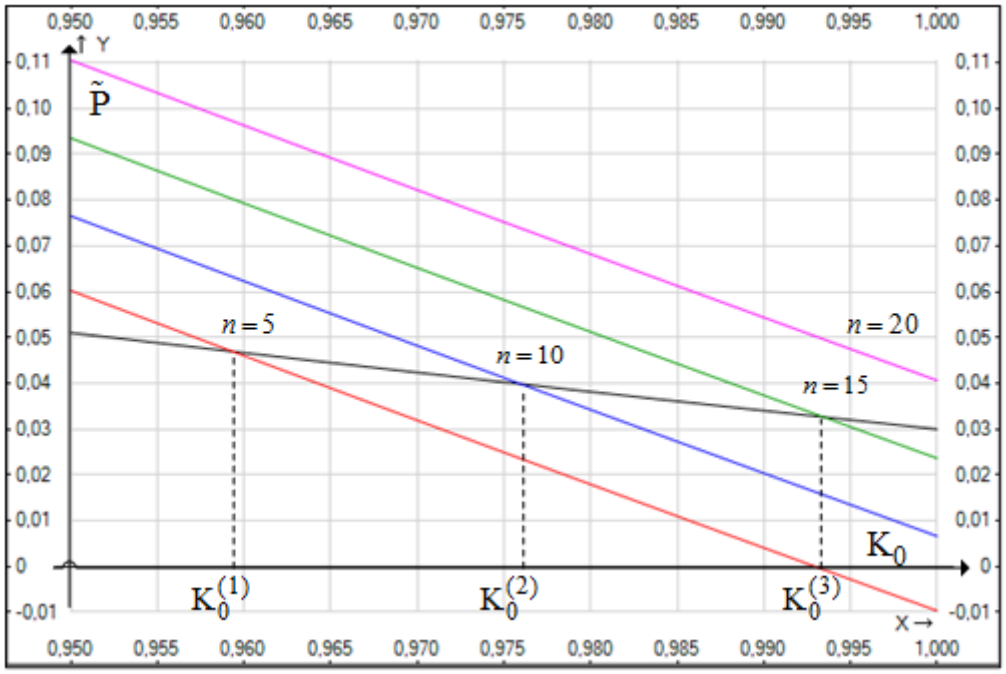


Рис. 3.33 Залежність зміни ефективності систем від надійності КеОМ і локальних регуляторів на мікроконтролерах

З графіків виходить, що із зростанням коефіцієнта готовності втрати в ієрархічній децентралізованій системі в порівнянні з централізованою зменшуються ($K_0^{(1)}, K_0^{(2)}, K_0^{(3)}$ - точки на графіці, де втрати в обох системах однакові). З (3.40) витікає, що втрати в ієрархічній системі лінійно залежать від числа і вартості локальних мікроконтролерних регуляторів. При заданих показниках надійності (K_0, α) і вартості підсистем в технічній структурі системи управління ($C_0^{ц}, C_0^{дц}$ и c_n) число контурів локального управління *n* може бути вибране на підставі рішення рівнянь (3.39) і (3.40).

Якщо порівняти (3.39) і (3.40), та ввести нормалізовані коефіцієнти

$$a_1 = 10^{-3}\delta_2 B^*;\ a_2 = 10^{-3}c_n;\ a_3 = 10^{-3}(C_0^{\text{дц}} - C_0^{\text{ц}}), \qquad (3.41)$$

можемо отримати необхідні рівняння двох змінних K_0 та n:

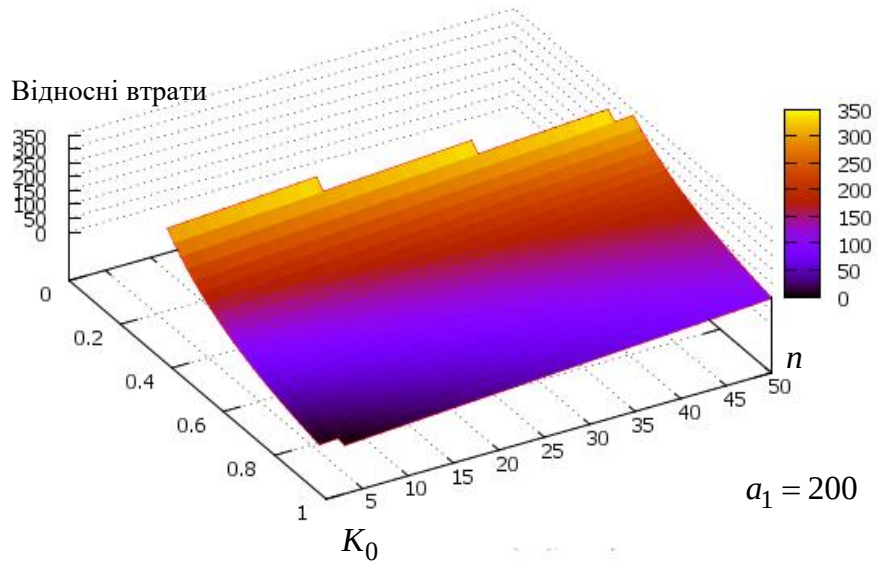
$$a_1(1-K_0)/K_0 + a_2 n + a_3 = 0.$$

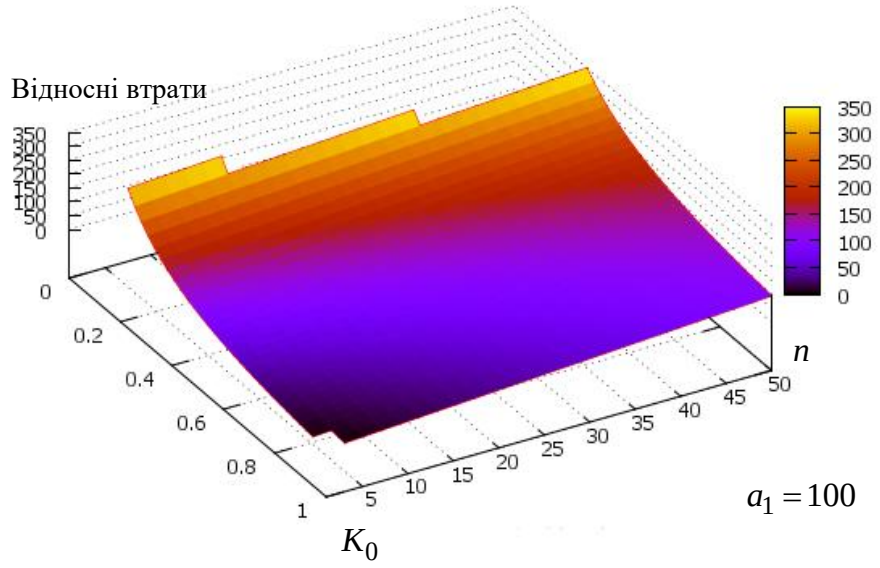
На рисунку 3.34 представлені площини, які відображають геометричне місце точок $K_0^{(i)}$, в яких втрати в централізованій і ієрархічній (децентралізованій) системах однакові і відповідає зміні пріоритетів в побудові систем в залежності від коефіцієнтів готовності і кількості локальних регуляторів. Аналіз отриманих результатів показує, що раціональні рішення розташовані в зоні високих коефіцієнтів готовності і невеликої кількості локальних підсистем. Зона раціональних рішень збільшується, якщо зменшується значення коефіцієнту a_1. Кількість локальних контурів не перевершує 15 підсистем.

Слід враховувати, що отримані результати справедливі тільки для випадку кінцевих і адитивних втрат. Для об'єктів, повна відмова пристроїв автоматики яких може призвести до аварій типу катастроф або пов'язана з можливістю людських жертв, вибір безсумнівно повинен бути вирішений на користь комбінованої системи.

3.5.2 Метод вибору технічних структур цифрових керуючих систем за критерієм ефективного використання обчислювальних ресурсів

У сучасних і перспективних системах автоматизації керування транспортними процесами на кожному етапі їх розвитку вирішується питання декомпозиції системи та представлення її або у вигляді централізованої структури, або ієрархічної системи, для якої необхідно вибрати раціональну ступінь декомпозиції [69]. Складність цього завдання полягає в тому, що на ранніх стадіях проектування відсутні багато параметрів технічних засобів автоматизації, що вимагає





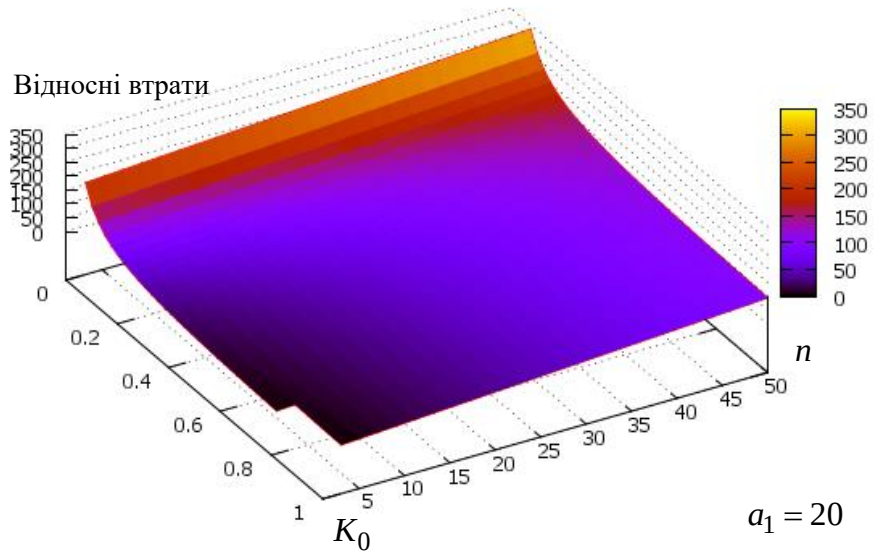


Рис. 3.34 - Площини рівних втрат централізованих і ієрархічних систем

введення низки часткових критеріїв і експертних оцінок. У роботі [150] розглядаються умови доцільності передачі в ієрархічній системі функцій місцевих пристроїв автоматики керуючим пристроям більш високого рівня, де використовуються багаторівневі оцінки збитку в залежності від режимів функціонування технологічної системи та відповідних відносних втрат, оцінки вартості центральних керуючих ЕОМ і мікроконтролерів в локальних регуляторах, а також інтенсивності відмов та інтенсивності відновлення використовуваних пристроїв. Розвитком описаного підходу являються моделі, запропоновані в [65], де одним з параметрів виступає коефіцієнт готовності і наводяться експертні оцінки втрат і вартостей технічних засобів. У роботі [189] пропонуються відносні оцінки і порівняння надійності централізованих і децентралізованих АСК ТП, де для кожного стану вводиться ваговий коефіцієнт втрат і розраховуються середні втрати. Імітаційно-аналітичний підхід пропонується в [213], а в [211] використовуються набори часткових якісних критеріїв і обмежень для вибору централізованої або децентралізованої системи автоматизації. Для вибору варіанта структури децентралізованої мікропроцесорної системи управління сортувальною гіркою в [116] вперше запропонований набір відносних коефіцієнтів ефективності збільшення вартості АСК по 5 частковим і одному узагальненому показникам.

Описані підходи не враховують складності автоматизованих процесів управління, які визначають необхідні обчислювальні ресурси системи і, в кінцевому підсумку, її вартість (див. рис. 3.35).

Необхідні обчислювальні ресурси зазвичай виражаються у вигляді сумарної обчислювальної потужності W (продуктивності, швидкодії в MIPS - у мільйонах команд в секунду) всіх комп'ютерів в системі. Для оцінки вартості обчислювальних ресурсів використовується відомий закон Гроша

$$C_0 = k_0 W^{g_1}, \qquad (3.42)$$

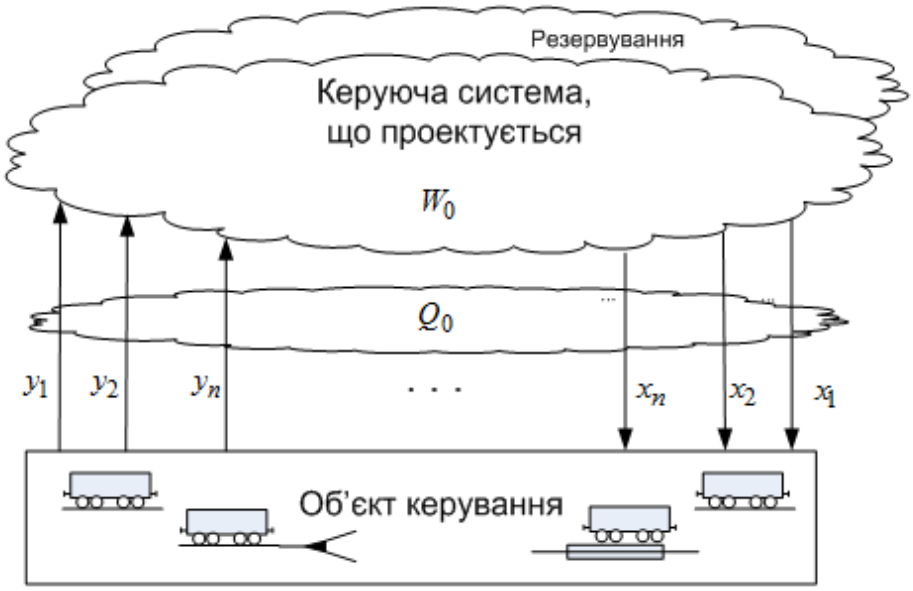


Рис. 3.35 Система, що проектується з потрібною обчислювальною потужністю W_0 і інформаційним потоками Q_0 (централізована структура)

де C_0 - вартість ресурсів (дол.), k_0 - коефіцієнт пропорційності, $g_1 = 0{,}5$, W - продуктивність в MIPS.

Незважаючи на солідний вік закону [190], він продовжує діяти [192], [197]. За твердженням Ейн-Дора [181] він добре працює в сімействі мікроконтролерів і керуючих ЕОМ (наприклад, Advantech [158]). У роботах [144, 182] пропонуються підходи до вибору концепції управління за допомогою мікроконтролерів і структури систем автоматизації на основі описаного закону.

У даній роботі, у розвитку [182], пропонується методика, аналітичні та графічні моделі і методи для вибору технічних структур автоматизації сортувальних гірок за інтегральним критерієм оцінки варіантів структур з урахуванням вартості потрібних обчислювальних і комунікаційних ресурсів, втрат на диспетчеризацію і зміни потоків в каналах, а також організацію резервування.

Розглянемо основні припущення. В даний час доведена на практиці перевага використання в АСКТП децентралізованих керуючих систем

ієрархічного типу в порівнянні з централізованим управлінням [211, 69, 144]. При цьому вважається наступне.

1) Цифрові управляючі системи (ЦКС) ієрархічного типу, що містять n спеціалізованих локальних мікроконтролерних підсистем F_i, $i=\overline{1,n}$ і одну універсальну підсистему верхнього рівня F_{11}, є більш живучими в порівнянні з централізованою системою (рис. 3.36).

2) Попередня обробка даних в локальних підсистемах скорочує потоки інформації в одиницю часу між рівнями в керуючій системі до значення $Q_{\text{дцп}}$.

$$Q_0 = Y \cup X,\ Y = \{y_1, y_2, ..., y_n\},\ X = \{x_1, x_2, ..., x_n\}, \tag{3.43}$$

$$Q_{\text{дцп}} = \tilde{Y} \cup \tilde{X},\ \tilde{Y} = \{\tilde{y}_1, \tilde{y}_2, ..., \tilde{y}_n\},\ \tilde{X} = \{\tilde{x}_1, \tilde{x}_2, ..., \tilde{x}_n\}, \tag{3.44}$$

$$Q_{\text{дцп}} < Q_0, \tag{3.45}$$

$$W_0 = W_{11} + \sum_{i=1}^{n} W_i. \tag{3.46}$$

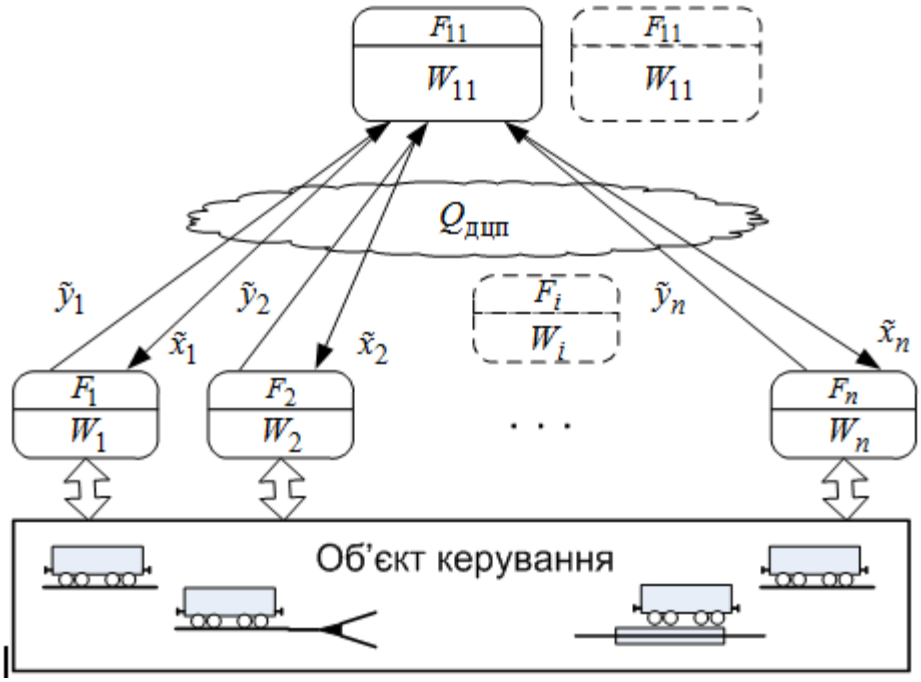


Рис. 3.36 Децентралізована ієрархічна ЦКС

3) Наявність локальних підсистем знижує навантаження на керуючу ЕОМ верхнього рівня не тільки за рахунок перерозподілу функцій, але і за

рахунок зменшення витрат на диспетчеризацію завдань.

4) У децентралізованій системі повинні використовуватися уніфіковані проектні рішення, завдяки чому скорочуються витрати на проектування і експлуатацію таких систем.

Значення характеристик W_0 *і* Q_0 можна визначити на основі аналізу інформаційної моделі функцій, що автоматизуються, у відповідності з методикою [78].

Розглянемо моделі для порівняння структур ЦКС за критерієм витрат на обробку даних. В якості основної альтернативи централізованої системи управління (рис. 3.35) будемо розглядати структуру, представлену на рис. 3.36.

Потрібна обчислювальна потужність W_0 для управління об'єктом автоматизації розподіляється між рівнями управління за допомогою коефіцієнта децентралізації $k_д$. Для центральної підсистеми $W_{11} = k_д W_0$, для всіх *n* підсистем нижнього рівня - $(1-k_д)W_0$.

Витрати на підсистему залежать від її обчислювальної потужності і обчислюються за степеневим законом

$$C_0 = k_0 W_0^{g_1}; \;\; g_1 = 0,5. \qquad (3.47)$$

Вартісний коефіцієнт k_0 залежить від типу розв'язуваних завдань (науково-технічні завдання або економічні завдання бізнес-систем) і перебуває в межах $k_0 = 1 \pm 0,005$ [197]. Для технічних систем в цілому $0,5 \le g_1 \le 0,8$, а для обчислювальних машин - $g_1 = 0,5$ [212].

Для порівняльної оцінки структур будемо використовувати співвідношення або показник ефективності обробки даних

$$E_о = \frac{C^{дц}}{C^{ц}}. \qquad (3.48)$$

Тут, для централізованої системи,

$$C^{ц} = k_1 W_0^{g_1}. \qquad (3.49)$$

Для децентралізованої системи потужність підсистеми верхнього рівня буде складати $k_{\text{д}}W_0$, а інша потужність $(1-k_{\text{д}})W_0$ буде рівномірно розподілена між n підсистемами нижнього рівня як $(1-k_{\text{д}})k_{\text{д}i}W_0$. Тоді отримаємо

$$C^{\text{дц}} = k_1(k_{\text{д}}W_0)^{g_1} + \sum_{i=1}^{n} k_2[(1-k_{\text{д}})k_{\text{д}i}W_0]^{g_1}. \qquad (3.50)$$

Передбачається, що k_1 для центральних підсистем більше, ніж k_2 для децентралізованих підсистем. Оскільки потужність між n підсистемами нижнього рівня розподіляється рівномірно, то $k_{\text{д}i} = \frac{1}{n}$, $i = \overline{1,n}$.

Прийнявши, що $g_1 = 0,5$; $k_{21} = \frac{k_2}{k_1}$ і підставивши (3.49) і (3.50) в (3.48) отримаємо

$$E_{\text{о}} = \frac{C^{\text{дц}}}{C^{\text{ц}}} = k_{21}\sqrt{n}\sqrt{1-k_{\text{д}}} + \sqrt{k_{\text{д}}}. \qquad (3.51)$$

Позначивши $k_{21}\sqrt{n} = y$, отримаємо графік площини, яка розділяє області переваги децентралізованих-централізованих структур ЦКС (рис. 3.37). Вважаючи, що $k_{\text{д}}$ - мале, тобто основне навантаження переноситься на нижній рівень ЦКС, і $k_{21}\sqrt{n}$ - невелике, отримуємо $E_0 < 1$ - децентралізація краще. При зменшенні попередньої обробки $(k_{\text{д}} \uparrow)$ $E_0 \approx 1$. При великій кількості підсистем у децентралізованій ЦКС $(k_{21}\sqrt{n} \uparrow)$ - $E_0 > 1$. У той же час перенесення всієї обробки на нижній рівень $(k_{\text{д}} \downarrow)$ призводить до зростання витрат.

На рисунку 3.38 наведена гранична крива рівних витрат, при якій $E_0 = 1$. У цьому випадку (3.51) можна перетворити в

$$y = \frac{1-\sqrt{k_{\text{д}}}}{\sqrt{1-k_{\text{д}}}}. \qquad (3.52)$$

Отримана крива поділяє області переваги структур ЦКС за витратами на обробку даних.

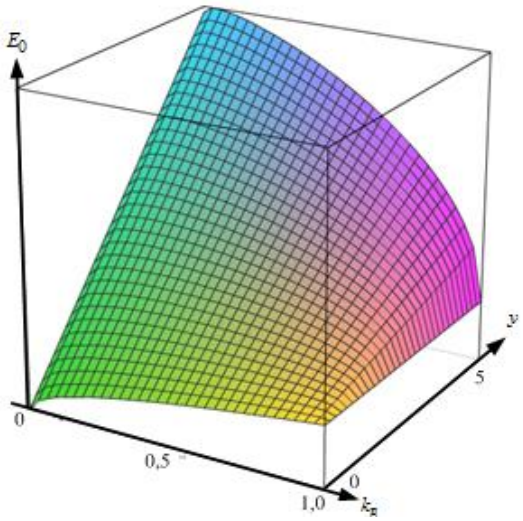


Рис. 3.37 Визначення області переваги децентралізованої обробки даних ($E_0 < 1$)

Рис. 3.38 Області переваги структур обробки даних

Тепер розглянемо моделі порівняння структур ЦКС за умовою сумарних витрат на обробку та передачу даних.

У попередній моделі враховувалися тільки витрати на обробку даних. Однак ми відзначали, що змінюються і потоки даних між рівнями в ЦКС (див. рис. 3.36) і відповідні витрати. Витрати на передачу даних у централізованих ЦКС можна оцінити таким чином [210]:

$$C_{\text{п}}^{\text{ц}} = \beta Q_0^{g_2}. \qquad (3.52)$$

Для децентралізованих систем

$$C_{\text{п}}^{\text{дц}} = \beta(f_{\text{дцп}} Q_0)^{g_2} = \beta(Q_{\text{дцп}})^{g_2}. \qquad (3.53)$$

В (3.52) і (3.53) - β, g_2 постійні коефіцієнти, $f_{\text{дцп}}$ - коефіцієнти пониження інформаційних потоків при переході на децентралізовану обробку. Надалі будемо вважати, що $f_{\text{дцп}} = 0{,}1$, тобто обсяги потоків інформації зменшуються в 10 разів. Крім того, введемо коефіцієнт

$$k_{\text{п/о}}^{\text{ц}} = \frac{\beta Q_0^{g_2}}{k_0 W_0^{g_1}}, \qquad (3.54)$$

який описує відношення витрат на потужності передачі до витрат на потужності обробки в централізованій ЦКС. За даними [144], для централізованої системи $k_{\text{п/о}}^{\text{ц}} = 4$. В результаті, припускаючи, що $g_1 = g_2 = 0{,}5$, і ґрунтуючись на раніше прийнятих припущеннях, можна записати

$$E_{\text{о/п}} = \frac{C^{\text{дц}} + C_{\text{п}}^{\text{дц}}}{C^{\text{ц}} + C_{\text{п}}^{\text{ц}}} = \frac{k_{21}\sqrt{n}\sqrt{1-k_{\text{д}}} + \sqrt{k_{\text{д}}} + k_{\text{п/о}}^{\text{ц}}\sqrt{f_{\text{дцп}}}}{1 + k_{\text{п/о}}^{\text{ц}}} =$$

$$= \frac{y\sqrt{1-k_{\text{д}}} + \sqrt{k_{\text{д}}} + 4\sqrt{0{,}1}}{1+4} = 0{,}2y\sqrt{1-k_{\text{д}}} + 0{,}2\sqrt{k_{\text{д}}} + 0{,}25. \qquad (3.55)$$

Графік отриманої функції наведений на рис. 3.39.

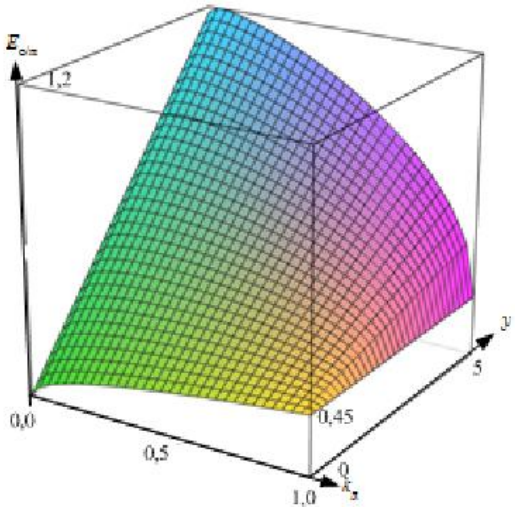


Рис. 3.39 Залежність ефективності застосування структур за критерієм витрат на обробку та передачу даних при $k_{\text{п/о}}^{\text{ц}} = 4,\ f_{\text{дцп}} = 0{,}1$

При $E_{\text{о/п}}=1$, $y=k_{21}\sqrt{n}$, $k^{\text{ц}}_{\text{п/о}}=4$ отримаємо наступну залежність y від $k_{\text{д}}$ і $f_{\text{дцп}}$:

$$y=\frac{5-\sqrt{k_{\text{д}}}-4\sqrt{f_{\text{дцп}}}}{\sqrt{1-k_{\text{д}}}}. \qquad (3.56)$$

Графік площини, що розділяє області переваги централізованих-децентралізованих структур наведено на рис. 3.40. Область над площиною відповідає кращим варіантам централізованих ЦКС, під площиною - децентралізованим системам. при зниженні $f_{\text{дцп}}$ від 1 до 0 межі децентралізованих систем розширюються. При незначному зменшенні $f_{\text{дцп}}$ в області $k_{\text{д}}=1$ діапазон для децентралізованих систем різко знижується. Таким чином, розширенню області децентралізованих систем сприяють зменшення y (або n), збільшення $k_{\text{дцп}}$ і зниження $f_{\text{дцп}}$.

Далі пропонуються моделі порівняння структур ЦКС за умовою сумарних витрат на обробку даних з диспетчеризацією.

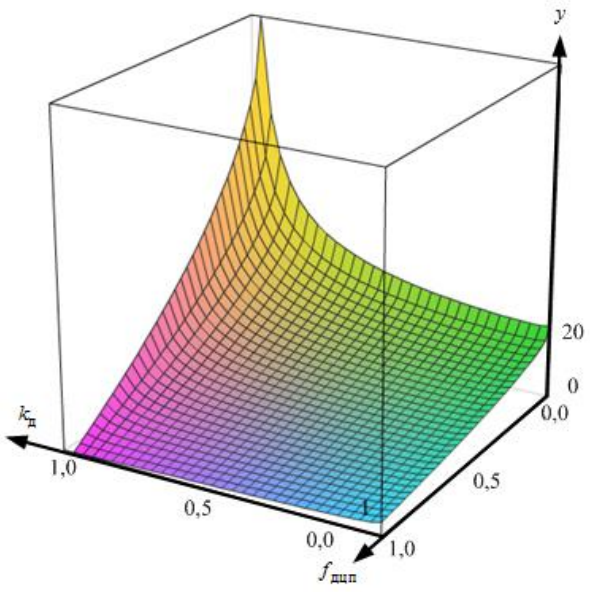


Рис. 3.40 Гранична площина між областями уподобання централізованих-децентралізованих ЦКС

У процесі вибору структури ЦКС важливо враховувати не тільки зміни в структурній організації систем, але і в алгоритмах їх функціонування. Це пов'язано в першу чергу з вирішенням завдань диспетчеризації і побудовою багаторівневої системи переривань в централізованій системі, що збільшує її вживану обчислювальну потужність. Будемо її враховувати за допомогою коефіцієнта $f_{\text{дисп}}$, який назвемо фактором диспетчеризації. Тоді

$$C^{\text{ц}}_{\text{дисп}} = k_1 (f_{\text{дисп}} W)^{g_1}, \;\; f_{\text{дисп}} \geq 1. \tag{3.57}$$

Відповідно до методики, що застосовується, обчислюємо коефіцієнт відношення витрат (використовуючи (3.50) і (3.57):

$$E_{\text{дисп}} = \frac{C^{\text{дц}}_{\text{о}}}{C^{\text{ц}}_{\text{дисп}}} = \frac{k_{21}\sqrt{n}\sqrt{1-k_{\text{д}}} + \sqrt{k_{\text{д}}}}{\sqrt{f_{\text{дисп}}}}. \tag{3.58}$$

Графік залежності (3.58) представлений на рис. 3.41. Гранична площина між областями переваги централізованих-децентралізованих ЦКС, коли $E_{\text{дисп}} = 1$, описується виразом

$$y = \frac{\sqrt{f_{\text{дисп}}} - \sqrt{k_{\text{д}}}}{\sqrt{1-k_{\text{д}}}}. \tag{3.59}$$

Графік функції (3.59) зображений рис. 3.42.

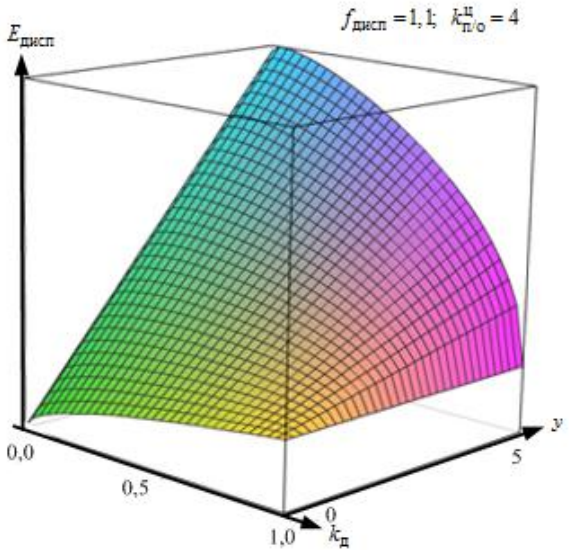


Рис. 3.41 Врахування диспетчеризації при виборі структури ЦКС

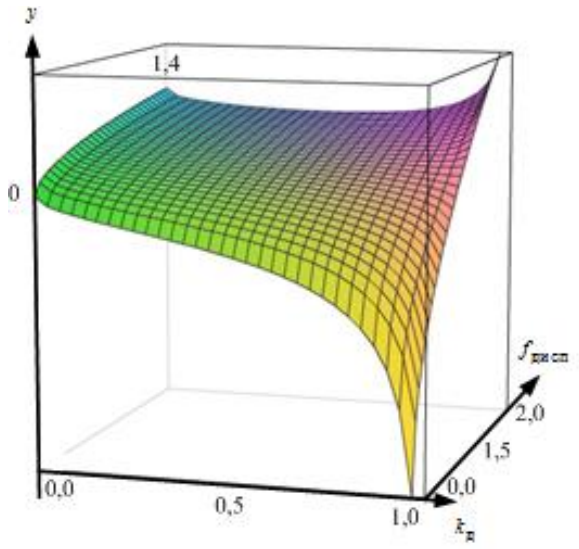


Рис. 3.42 Вплив витрат диспетчеризації на розподіл областей переваги структур ЦКС

Наведені графіки показують область переваги децентралізованих структур за рахунок необхідності додаткової потужності на організаційні процедури в центральній керуючій ЕОМ.

Моделі порівняння структур ЦКС повинні враховувати і витрати на резервування підсистем. Відомо, що для систем, що працюють у реальному масштабі часу, пред'являються високі вимоги до показників надійності. До таких систем відносяться і автоматизовані системи управління процесами розформування-формування поїздів. Як правило, в системах використовується гаряче резервування. Наприклад, в системі АСКРСГ (станція Ясинувата), побудованої на базі міні-ЕОМ СМ-2М [69], центральна машина дублювалася з комутацією сигналів на рівні пристроїв зв'язку з об'єктом (ПЗО) (див. рис. 3.35). У децентралізованих системах на основі мікропроцесорних промислових ЕОМ зазвичай резервують (дублюють) центральну, координуючу ЕОМ і для підсистем нижнього рівня виділяють один комплекс з можливістю його використання як ковзний резерв. На рисунку 3.36 резервні підсистеми показані пунктирною лінією. Будь-яке резервування пов'язано з додатковими ресурсними витратами, що безумовно впливає на вибір відповідної структури.

Введемо коефіцієнт ступеня резервування $k_{\text{рез}}$, який приймає цілі значення з області натуральних чисел □, починаючи з 2 ($k_{\text{рез}} \geq 2,\ k_{\text{рез}} \in$ □).

Тоді витрати для централізованих систем будуть складати

$$C_{\text{рез}}^{\text{ц}} = k_{\text{рез}} * C_{\text{дисп}}^{\text{ц}} = k_{\text{рез}} k_1 (f_{\text{дисп}} W)^{g_1}. \qquad (3.60)$$

При дублюванні $k_{\text{рез}} = 2$. Будемо припускати, що для децентралізованої структури використовується той же коефіцієнт, тобто дублюється центральна, координуюча підсистема і будь-яка з n локальних підсистем спеціальної резервної підсистемою. У цьому випадку витрати на резервування в децентралізованій системі можна оцінити за такою формулою

$$C_{\text{рез}}^{\text{дц}} = k_{\text{рез}} * \left[k_2 ((1 - k_{\text{д}}) \frac{W}{n})^{g_1} + k_1 (k_{\text{д}} W)^{g_1} \right]. \qquad (3.61)$$

Ефективність організації резервування за критерієм мінімальної вартості будемо визначати коефіцієнтом

$$E_{\text{рез/о}} = \frac{C_{\text{рез}}^{\text{дц}}}{C_{\text{рез}}^{\text{ц}}} = \frac{k_{21}}{\sqrt{n}} * \sqrt{1 - k_{\text{д}}} + \sqrt{k_{\text{д}}}. \qquad (3.62)$$

Виконаємо перетворення $\frac{k_{21}}{\sqrt{n}} = \frac{k_{21}\sqrt{n}}{n} = \frac{y}{n}$. Тоді площина, яка розділяє області ефективності, буде описуватись виразом

$$\frac{y}{n} * \sqrt{1 - k_{\text{д}}} + \sqrt{k_{\text{д}}} = 1. \qquad (3.63)$$

Звідки

$$y = \frac{n - \sqrt{k_{\text{д}}}}{\sqrt{1 - k_{\text{д}}}}. \qquad (3.64)$$

На рис. 3.43 представлений графік площини, що розділяє області переваги резервування в централізованих - децентралізованих системах.

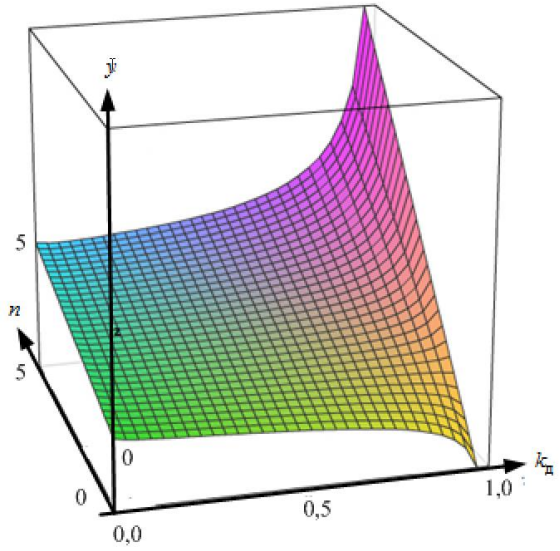


Рис. 3.43 Площина, що розділяє область переваги централізованих (над площиною) і децентралізованих (під площиною) систем

Запропоновані моделі можна використовувати окремо при дослідженні впливу окремих характеристик системи на відповідні втрати і структури. Але найбільш корисним може бути інтегральний показник ефективності структур ЦКС.

На підставі описаних часткових показників порівняльної оцінки централізованих-децентралізованих ЦКС (3.49), (3.50), (3.52), (3.53), (3.57), (3.60), (3.61) пропонується комплексний або інтегральний показник, що враховує процеси обробки і передачі даних, диспетчеризації заявок у централізованих системах та організацію резервування.

$$E_{\text{инт}} = \frac{C^{\text{дц}} + C_{\text{п}}^{\text{дц}} + C_{\text{рез}}^{\text{дц}}}{C_{\text{дисп}}^{\text{ц}} + C_{\text{п}}^{\text{ц}} + C_{\text{рез}}^{\text{ц}}}. \qquad (3.65)$$

Зробивши підстановки і для рівняння площини $E_{\text{инт}} = 1$ отримаємо наступний вираз

$$(1+k_{\text{д}})(\sqrt{k_{\text{д}}} - \sqrt{f_{\text{дисп}}}) + k_{\text{п/о}}^{\text{ц}}(\sqrt{f_{\text{дцп}}} - 1) + k_{21}\sqrt{1-k_{\text{д}}}(1 + \frac{k_{\text{рез}}}{\sqrt{n}}) = 0. \quad (3.66)$$

По (3.66) можна побудувати площину, що розділяє області переваги централізованих-децентралізованих ЦКС, задавши набір параметрів системи. Наприклад, якщо прийняти, що $f_{\text{дисп}} = 1{,}1$, $f_{\text{дцп}} = 0{,}9$, $k_{\text{рез}} = 2$, $k^{\text{ц}}_{\text{п/о}} = 1$ і $k_{21} = 1$, графік буде мати вигляд, представлений на рис. 3.44.

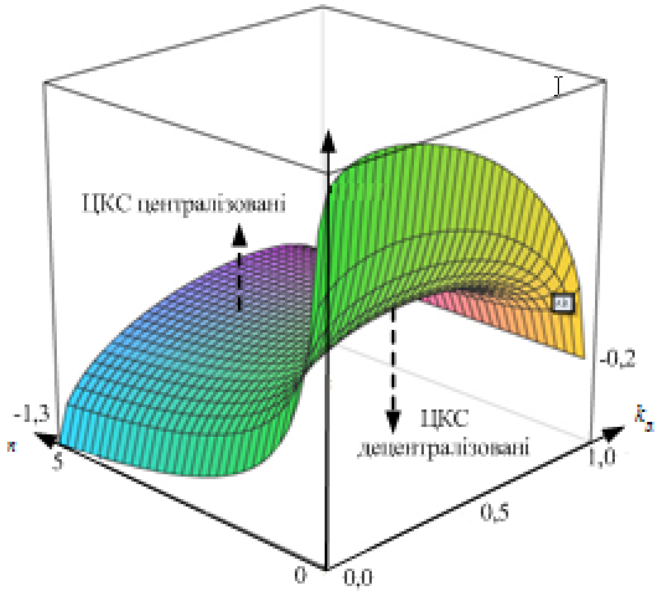


Рис. 3.44 Інтегральна площина, що розділяє області переваги централізованих і децентралізованих ЦКС

Отримана залежність дозволяє визначити на ранніх стадіях проектування ЦКС при заданих параметрах функціонування область переваги децентралізованих систем, яка збільшується при зниженні кількості підсистем на нижньому ярусі n і зменшенні їх потужності ($k_{\text{д}} \to 1$).

Запропонований метод і набір аналітичних моделей дозволяють на стадіях системотехнічного проектування та модернізації ЦКС обґрунтувати вибір централізованої або ієрархічної (децентралізованої) структури.

3.5.3 Ресурсоощадний підхід до оцінювання витрат на побудову цифрових розподілених керуючих систем

У п. 3.5.1, 3.5.2 пропонуються дві методики оцінки та вибору варіантів побудови розподілених цифрових керуючих систем (ЦКС) на прикладі автоматизованої системи управління розформуванням-формуванням складів на сортувальних гірках. У цих роботах недостатньо уваги приділяється обліку витрат на засоби обміну даними в досліджуваних варіантах. В [65] ці витрати віднесені до витрат на мікроконтролерні підсистеми, а в [76] вони задаються коефіцієнтом від витрат на пристрої обробки даних і враховують швидкості передачі даних. У роботі [144] наведені експериментальні дані по витратах на обмін даними, припадає на централізовану систему і спосіб перерахунку цих витрат при кільцевої мережі. Хоча з точки зору мінімізації витрат на комунікації на об'єкті оптимальний результат дає побудова мінімального кістяка. У даній частині пропонується аналітична модель для оцінки витрат на обробку та обмін даними з мінімальною сумарною довжиною каналів передачі даних між підсистемами.

Основні припущення та початкові дані.

Припустимо, що є розподілена ЦКС, що складається з n локальних підсистем. Кожна ЕОМ обслуговує m точок автоматизації. Для сортувальних гірок, залежно від колійного розвитку, кількість таких точок коливається від 321 до 850 [105] На рисунку 3.45 показані централізована система (а), розподілена-кільцева (б) і система з мінімальним кістяковим деревом комунікацій (в).

Будемо вважати, що загальна довжина сполучного кабелю L_0 (лінії зв'язку до m локальних точок автоматизації l_i і магістральні лінії зв'язку $l_k^{\text{м}}$), приведена до загальної довжини L_1 в централізованій системі, коли n=1 (рис. 3.45, (а)). Тоді при заданому m і вибраному n можна отримати

$$L_0/L_1 = f(n,m).$$

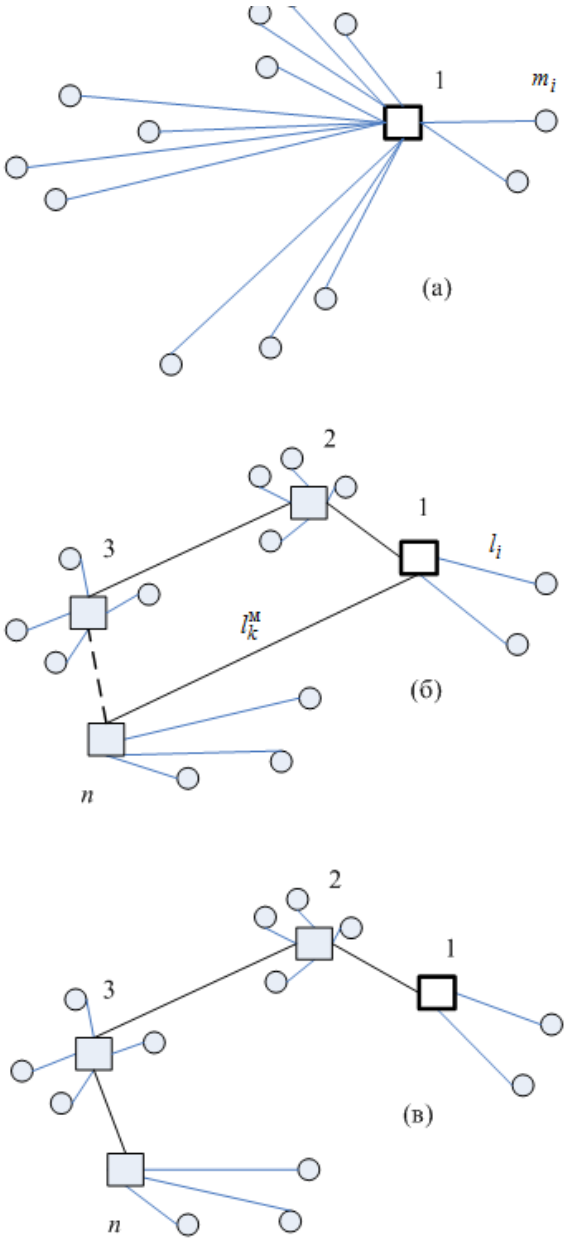


Рис. 3.45 Варіанти структур ЦКС

Таблиця 3.2

Значення функції $L_0/L_1 = f(n,m)$ **[144]**

	n						
	1	3	5	7	9	11	13
При $m=144$ $L_0/L_1 = f(n,m)$	1,00	0,64	0,50	0,44	0,39	0,36	0,34
При $m=3600$ $L_0/L_1 = f(n,m)$	1,00	0,62	0,43	0,34	0,30	0,27	0,26

Вартість витрат на централізовану систему (n = 1) складає

$$C^{ц} = C_1 = C_1^{обр} + C_1^{каб}. \qquad (3.67)$$

За експериментальними даними встановлено [144], що

$$C_1^{\text{обр}} / C_1 = 0,2 \quad \text{і} \quad C_1^{\text{каб}} / C_1 = 0,8 . \tag{3.68}$$

Будемо вважати, що при переході від централізованої структури до розподіленої системи з n контурами управління, сумарна потрібна обчислювальна потужність W_0 рівномірно розподіляється між підсистемами, тобто розподілена ЦКС є однорідною

$$W_0 = nW_i \quad \forall i \in \overline{1,n} . \tag{3.69}$$

При цьому

$$C_0^{\text{обр}} = aW_0^{g_1}, \text{ где } g_1 = 0,5 . \tag{3.70}$$

Визначення довжини кабелю в реальних системах залежить від множини факторів, тому точні розрахунки практично неможливі. Координати місця точок автоматизації та розміщення мікроконтролерних підсистем можна визначити тільки після впровадження ЦКС. Тим більше прокладання кабелю може вестися по збудованим існуючим та новим каналам. Для сортувальної гірки можна вважати, що розподіл цих точок є рівномірним.

На підставі прийнятих припущень і (3.68), (3.69), (3.70) можна записати відносні витрати на мікроконтролери в розподіленій системі, приведені до витрат в централізованій структурі

$$C_n^{\text{обр}} = n * C_i = n * a * \sqrt{\frac{W_0}{n}} = \frac{n * a * \sqrt{W_0}}{\sqrt{n}} = \sqrt{n} * C_1^{\text{обр}} . \tag{3.71}$$

Взявши відношення, отримаємо

$$\frac{C_n^{\text{обр}}}{C_1} = \frac{\sqrt{n} * C_1^{\text{обр}}}{C_1} = 0,2\sqrt{n} . \tag{3.72}$$

Відносну вартість кабельних з'єднань можна визначити наступним чином:

$$\frac{C_n^{\text{каб}}}{C_1} = \frac{C_n^{\text{каб}}}{C_1^{\text{каб}}} * \frac{C_1^{\text{каб}}}{C_1} = \frac{C_n^{\text{каб}}}{C_1^{\text{каб}}} * 0{,}8 \approx \frac{L_0}{L_1} * 0{,}8 = 0{,}8\, f(n,m). \qquad (3.73)$$

Приймаючи (3.73), перетворимо таблицю 3.2 в таблицю 3.3.

Таблиця 3.3

Значення функції $0{,}8\, f(n,m)$

	n						
	1	3	5	7	9	11	13
При $m = 144$ $L_0/L_1 = f(n,m)$	0,80	0,51	0,40	0,35	0,31	0,29	0,27
При $m = 3600$ $L_0/L_1 = f(n,m)$	0,80	0,50	0,34	0,27	0,24	0,22	0,21

Для вирішення завдання визначення кількості підсистем у розподіленій структурі за критерієм мінімізації витрат на мікроконтролери та лінії зв'язку можна запропонувати графічний підхід. На рис. 3.46 представлені графіки відповідних витрат $\frac{C_n^{\text{каб}}}{C_1}$; $\frac{C_n^{\text{обр}}}{C_1}$ і $\frac{C_n^{\text{каб}} + C_n^{\text{обр}}}{C_1}$.

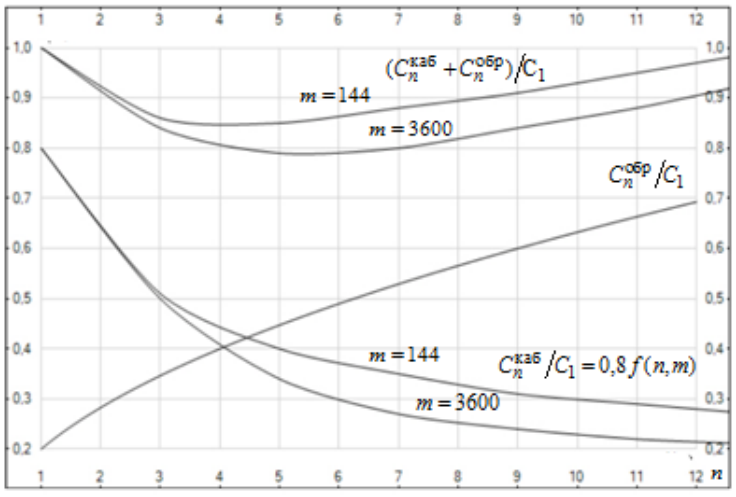


Рис. 3.46 Графічні визнаення $\min\left\{(C_n^{\text{каб}} + C_n^{\text{обр}})/C_1\right\}$

Отримані результати показують, що мінімум витрат відповідає 4-5 підсистемам (n = 4-5), тобто в ієрархічній структурі одна координуюча підсистема - на верхньому рівні, і на нижньому рівні - 3 або 4 технологічні підсистеми. Ці результати підтверджують отримані дані за допомогою інших аналітичних моделей [65, 76], що підтверджує коректність побудови моделей і можливість їх практичного застосування в процесі проектування нових і модернізації існуючих ЦКС на сортувальних гірках.

Отримані моделі можуть використовуватися на стадії системотехнічного проектування цифрових керуючих систем після аналізу об'єкта автоматизації, синтезу варіантів інформаційно-керуючої структури (визначення кількості підсистем) і визначення потрібної сумарної продуктивності ЦКС (за методикою проектування систем автоматизації сортувальних гірок п. 2.4.4 [78]). В якості розрахункової структури рекомендується використовувати не кільцеву структуру, а деревоподібну, що представляє мінімальне кістякове дерево. У цьому випадку будуть потрібні зміни в таблицях 3.2 і 3.3. Запропоновані моделі дозволять підвищити ефективність використання обчислювальних і кабельних ресурсів ЦКС і скоротити час їх проектування.

РОЗДІЛ 4

МОДЕЛІ ТА МЕТОДІВ РОЗРАХУНКУ ХАРАКТЕРИСТИК КОМП'ЮТЕРНИХ СИСТЕМ КЕРУВАННЯ РЕАЛЬНОГО ЧАСУ

4.1 φ-транзакція як основна модель для оцінки інформаційних і часових характеристик сервіс-орієнтованих систем

4.1.1 Визначення поняття φ-транзакція

Для дослідження і проектування інформаційно-керуючих систем реального масштабу часу сортувальних станцій введемо поняття φ-транзакція (англ. transaction, від лат. transactio — угода, договір).

φ-транзакція - це логічно об'єднана послідовність функціонально-алгоритмічних і програмних блоків операцій (далі - ФПБ), яка обробляється цілком від моменту появи в керуючій системі події, що вимагає обробки, до моменту завершення її обробки з видачею повідомлення і/або керуючої дії на пристрої об'єкту керування.

Для систем керування сортувальними станціями множина подій описується як $E = E^{w} \bigcup E^{sys} \bigcup E^{timer}$ (див. п. 2.2), а вся система задається вісімкою (2.5)

$$A = \left(T, Q, E, Y, \phi, \beta, f_{\mu}, f_{\mu}^{\max} \right),$$

в якій введено $f_{\mu}^{\max}$ як множина граничних значень функції переходів $f_{\mu} \le f_{\mu}^{\max}$.

φ-транзакція повинна володіти властивостями ACID [125]. Це абревіатура чотирьох слів, що означають атомарність (atomicity), узгодженість (consistency), ізоляцію (isolation) і стійкість (durability).

Атомарність. Атомарність представляє одиницю роботи. У відношенні

φ- транзакції це означає, що або вся одиниця роботи буде успішно виконана, або нічого не буде змінено.

Узгодженість. Стани перед стартом φ- транзакції і після її завершення повинні бути коректними. Під час φ- транзакції стан може мати проміжні значення.

Ізоляція. Ізоляція означає, що φ-транзакції, які виконуються одночасно, ізольовані від стану, який змінюється в час φ-транзакції. φ-транзакція А не може бачити проміжного стану φ-транзакції В до тих пір, поки вона не буде завершена.

Стійкість. Після завершення φ-транзакції її результат повинен бути зафіксований на постійній основі. Це означає, що якщо відбудеться збій мікроконтролера або керуючої ЕОМ, то стан повинен бути відновлений після їх перезапуску.

Від відомих визначень поняття φ-транзакція відрізняється розширеним описом, необхідним для розрахунку характеристик АСК на ранніх стадіях її проектування, яке включає для множини Ф оброблюваних системою φ-транзакцій:

- опис використовуваних при виконанні φ-транзакцій основних масивів бази даних M ($\forall m_i \in M \ \{O_i\}$, де O_i, $i \in \overline{1,|M|}$ - розмір масиву або об'єм пам'яті в байтах);

- опис множини подій E, які вимагають реакції системи керування, тобто запуску φ-транзакцій ($\forall \varphi_j \in \Phi \ \{s_j, \lambda_j\}$, де $s_j \in S_o$ - множина вхідних сигналів і повідомлень, пов'язаних з відповідними подіями *E,* див. (2.24), λ_j - середня інтенсивність запуску j-ї φ-транзакції);

- опис множини умов і вимог успішного завершення φ-транзакцій $\forall \varphi_j \in \Phi \ \forall d_k \in D \ \{op_{kj}, Q_{kj}, K_{kj}^{\text{г}}, t_{kj}^{\text{гр}}\}$, де d_{kj} - k-й приймач інформації від j-ї φ-транзакції, $op_{kj} = \{$ видати повідомлення $msg_{kj} \in Msg^{out}$, видати керуючу

дію у вигляді сигналу $s_{kj} \in S^{out}$ (2.24), ... }, Q_{kj} - об'єм інформації в байтах, що передається *k*-му приймачу від *j*-ї φ-транзакції, коефіцієнт готовності і граничний час виконання φ-транзакції *j* для *k*-го приймача;

- опис кожного ФПБ, який входить до складу системи: $\forall \phi_l \in ФПБ \ \forall m_i \in M \ \forall com_n \in \square \ \{k_{i,l}, op_{i,l}, K_{nl}\}$, де $ФПБ = \{\phi_l \,|\, l \in \overline{1,|ФПБ|}\}$ множина ФПБ системи, $M = \{m_i \,|\, i \in \overline{1,|M|}\}$ - множина масивів бази даних, $\square = \{com_n \,|\, n = \overline{1,|\square|}\}$ - множина типів команд в суміші команд проектованої системи або мікс (mix), $0 \le k_{i,l} \le 1$ - коефіцієнт, що показує яка частина масиву *i* використовується при одному виконанні ФПБ *l*); $op_{j,l,i} = \{\text{чт,зп}\}$ - операції з пам'яттю *i* при виконанні *l*-го ФПБ в *j*-й φ-транзакції, K_{nl} - кількість команд типу *n* в *l*-му ФПБ.

Загальний вигляд φ-транзакцій і опис системи, що проектується, показані на рис. 4.1.

У наборі ФПБ, що входять в φ-транзакцію, виділяються вершини, в яких можливе множинне розгалуження процесу обробки даних (наприклад, ϕ3 має три варіанти продовження обчислень). В цьому випадку задається ймовірність вибору напряму розгалуження, сума яких дорівнює 1 (у прикладі, $p_{35} + p_{37} + p_{32} = 1$).

Всі описані характеристики φ-транзакцій використовуються в процесі проектування систем автоматизації сортувальних станцій відповідно до запропонованого підходу КСІ (рис. 5.4). Фактично множина φ-транзакцій (Ф) є М-модель системи, в якій виключаються ФПБ, що повторюються. φ-транзакції "розшивають", структурують М-модель по окремих сервісах (далі - заявкам), для яких специфіковані певні показники (умови) якості надання сервісу.

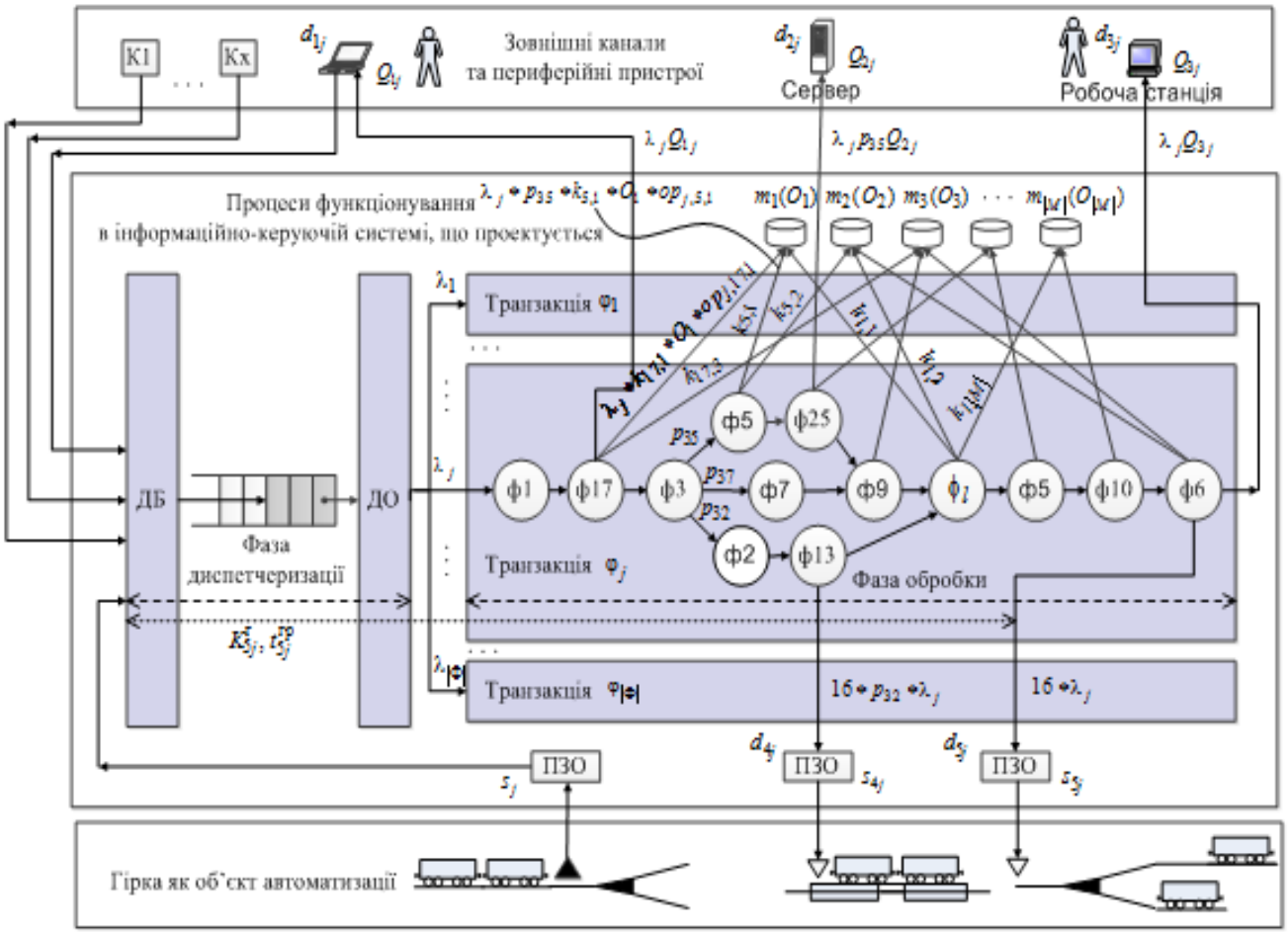


Рис. 4.1 φ-транзакції в системах автоматизації сортувальних станцій

4.1.2 Метод оцінки характеристик системи керування реального часу в процесі проектування

Початковими даними для розрахунку характеристик проектованої системи керування є:

- опис функціональних програмних блоків, який включає список ФПБ, використовувані ними масиви і коефіцієнти їх використання при кожному виконанні ФПБ, кількість операцій вибраних типів в кожному ФПБ (див. таблицю 4.1);

- опис φ-транзакцій у вигляді орієнтованих вершино і реброво-зважених графів з описом їх параметрів (див. рис. 4.1);

- опис вибраного заздалегідь типа мікропроцесора і часи виконання

операцій згаданих вище типів (приклади опису мікропроцесорів - в таблицях 4.2, 4.3 [139]).

Всі початкові дані представляються в табличному вигляді для подальшої обробки.

Слід зазначити, що ФПБ є або програмами (якщо даний блок узятий з бібліотеки програм розробника), або алгоритмами (для відомих алгоритмів управління гіркою), або функціями (для нових завдань). Це визначає точність оцінки K_{nl} и $k_{i,l}, op_{i,l}$.

Таблиця 4.1

Опис функціональних програмних блоків

Параметри	Функціональні програмні блоки					
	ϕ_1	ϕ_2	...	ϕ_l	...	$\phi_{\lvert ФПБ \rvert}$
Суміш команд (мікс) системи	Кількість операцій відповідного типу в ФПБ, тис. команд					
com_1	K_{11}	K_{12}		K_{1l}		$K_{1\lvert ФПБ \rvert}$
com_2	K_{21}	K_{22}		K_{2l}		$K_{2\lvert ФПБ \rvert}$
	...					
com_n	K_{n1}	K_{n2}		K_{nl}		$K_{n\lvert ФПБ \rvert}$
	...					
$com_{\lvert \square \rvert}$	$K_{\lvert \square \rvert 1}$	$K_{\lvert \square \rvert 2}$		$K_{\lvert \square \rvert l}$		$K_{\lvert \square \rvert \lvert ФПБ \rvert}$
Масиви і їх об'єми	Коефіцієнт використання масиву і тип операції					
$m_1(O_1)$	$k_{1,1}$, $op_{1,1}$	$k_{1,2}$, $op_{1,2}$		$k_{1,l}$, $op_{1,l}$		$k_{1,\lvert ФПБ \rvert}$, $op_{1,\lvert ФПБ \rvert}$
$m_2(O_2)$	$k_{2,1}$, $op_{2,1}$	$k_{2,2}$, $op_{2,2}$		$k_{2,l}$, $op_{2,l}$		$k_{2,\lvert ФПБ \rvert}$, $op_{2,\lvert ФПБ \rvert}$
	...					
$m_i(O_i)$	$k_{i,1}$, $op_{i,1}$	$k_{i,2}$, $op_{i,2}$		$k_{i,l}$, $op_{i,l}$		$k_{i,\lvert ФПБ \rvert}$, $op_{i,\lvert ФПБ \rvert}$
	...					
$m_{\lvert M \rvert}(O_{\lvert M \rvert})$	$k_{\lvert M \rvert,1}$, $op_{\lvert M \rvert,1}$	$k_{\lvert M \rvert,2}$, $op_{\lvert M \rvert,2}$		$k_{\lvert M \rvert,l}$, $op_{\lvert M \rvert,l}$		$k_{\lvert M \rvert,\lvert ФПБ \rvert}$, $op_{\lvert M \rvert,\lvert ФПБ \rvert}$

Таблиця 4.2

Час виконання арифметичних операцій, нс

Тип аргументу	int				double		
Операція	^	+	*	/	+	*	/
Pentium 4 2,5 ГГц	0,78	0,86	5,6	19	2	2,8	15
CoreDuo 1,8 ГГц	1,1	1,1	2,2	4,2	1,7	2,7	18
Pentium M 1,8 ГГц	1,1	1,1	2,2	8,7	1,7	2,7	17
Athlon 64X2 2,5 ГГц	1,4	1,4	1,6	17	1,8	1,8	8,3

Таблиця 4.3

Число тактів, необхідних для виконання інструкцій на процесорах Intel P4+

Інструкція	XOR	ADD	MUL	DIV	FADD	FMUL	FDIV (double)
Затримка обробки	0,5-1	0,5-1	3-18	22-70	5-6	7-8	38-40
Час обробки	0,5	0,5	1-5	23-30	1	2	38-40

Для вибору набору команд і отримання K_{nl} можна скористатися методикою "ПОЭТ" [105]. Для нових функцій існують характеристики типових алгоритмів, які рекомендуються для стадій системного проектування ІКОС (див. табл. 4.4) [37].

Таблиця 4.4

Характеристика типових алгоритмів

Клас алгоритмів (задач)	Вхідні параметри алгоритмів	Результівні характеристики		
		Внутрішня зв'язність	Число середніх операцій	Питома об'ємність
Первинна обробка вимірювань масштабування)	n - число вимірювань, m - розмір таблиці градуювання	$n+m$	nm	m
Статистичний аналіз	n - число вимірювань, m - число факторів, число коефіцієнтів	nm	nm^2	n
Оптимізація (лінійне прог. та ін.)	m - число змінних, n - число обмежень	$n(m+n)$	$n^2(m+n)$	n
Розрахунок техніко-економічних характеристик	n - число даних, m - число констант формули	$n+m$	nm	m

Для розрахунку швидкодії обчислювальних систем з метою їх порівняння і вибору використовується велика кількість різноманітних моделей, методів і бенчмарків (benchmark - эталонні тести Linpack, SPEC, SPECfp95, SPECweb97, TPC, WinMark, Winstone 97 і т.і.), але більшість з них орієнтовані на вже розроблені системи і їх практично неможливо використати на ранніх стадіях проектування [37, 121, 148].

У розробленій методиці (п. 5.4) пропонуються два способи оцінки швидкодії мікропроцесорів - середня швидкодія *i*-ї моделі на суміші завдань системи $\tilde{W}_i$ і середня швидкодія *i*-ї моделі на суміші φ-транзакцій системи керування, що проектується, $\tilde{W}_i^{\varphi}$.

Для автоматизованих систем керування сортувальною гіркою автором вже були визначені частотні суміші команд, вперше описані в роботі [83]. Проте, очевидно, що їх застосування доцільне тільки в тому випадку, якщо ми орієнтуємося на набір завдань системи АСКРСГ. Для врахування нових алгоритмів і програм керування потрібне коректування частотних характеристик сумішей. У якості рекомендується використовувати набори команд з таблиць 4.2 і 4.3 або методику експериментального програмування "ПОЭТ".

Швидкодія на частотній суміші команд при проектуванні системи для заздалегідь вибраного *i*-го типу МКП обчислюється таким чином:

$$\tilde{W}_i = \frac{1}{\sum_{n=1} \xi_n \tau_{ni}}, \qquad (4.1)$$

де ξ_n - частота або ймовірність появи операції n в процесі функціонування системи; τ_{ni} - середній час виконання команди n на мікропроцесорі *i*-го типу; одиницею вимірювання швидкодії є кількість операцій на суміші завдань системи за секунду - $\text{оп}/\text{с}$.

Значення ξ_n обчислюється як

$$\xi_n = \frac{K_n}{\sum_{h=1}^{|\square|} K_h}. \qquad (4.2)$$

Тут K_h - кількість команд *h*-го типу, які обробляються системою в процесі виконання φ-транзакцій.

K_h обчислюється за формулою (4.3) з урахуванням даних K_{nl} для кожного ФПБ з таблиці 4.1 і структури всіх φ-транзакцій, оброблюваних системою, з урахуванням інтенсивностей їх запуску, описаних на рис. 4.1.

$$\forall h \in |\square| \quad K_h = \sum_{\forall \varphi_j \in \Phi} \lambda_j * \sum_{\forall \phi_l \in \varphi_j} k_{lj} * K_{hl}. \qquad (4.3)$$

У формулі (4.3) $0 < k_{lj} \le 1$ - коефіцієнт, що враховує зниження кількості операцій в φ-транзакції *j* в ФПБ *l* за рахунок попередніх розгалужень і циклів (у зваженому графі). Цей коефіцієнт обчислюється в процесі перетворення структури і обчисленні характеристик φ-транзакцій (див. п. 4.2).

Отримана середня швидкодія *i*-ї моделі на суміші завдань майбутньої системи $\tilde{W}_i$ використовується в запропонованій методиці КСІ для порівняння різних типів МКП при їх виборі, а також для ресурсоощадних методів пошуку раціональних структур децентралізованих систем керування сортувальною гіркою (див. п. 3.3, 3.4).

Для сервіс-орієнтованих систем реального масштабу часу, в яких встановлено обмеження на час реакції або час обслуговування кожної φ-транзакції, пропонується обчислювати середню швидкодію *i*-ї моделі на суміші φ-транзакцій, або $\tilde{W}_i^{\varphi}$, яке вимірюється в кількості оброблюваних середньозважених φ-транзакцій за секунду ("$\varphi - \text{tps}$" - φ-transaction per second) або в хвилину - "$\varphi - \text{tpm}$". Для цього скористаємося виразом (4.4).

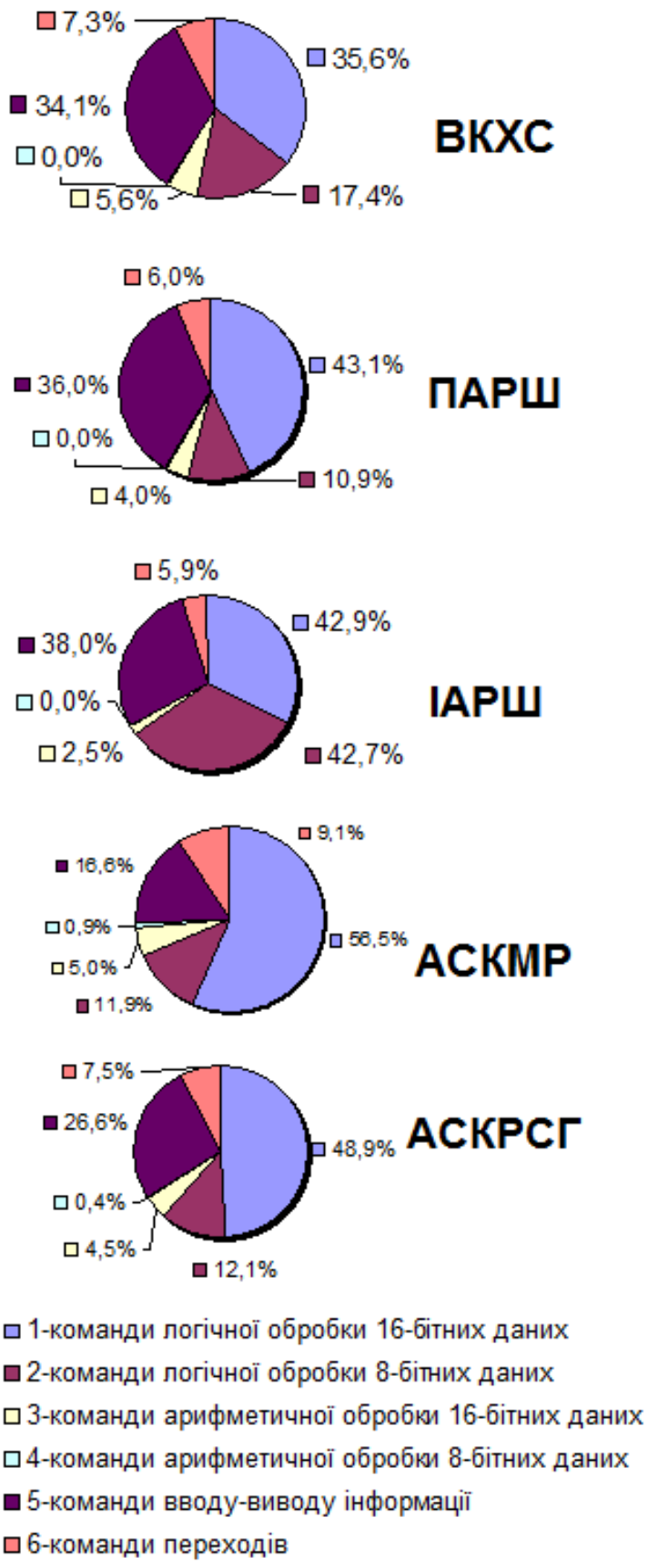


Рис. 4.2 Частотні суміші операцій основних функціональних підсистем і системи керування сортувальною гіркою

$$\tilde{W}_i^{\varphi} = \frac{\sum_{j=1}^{|\Phi|} \lambda_j}{\sum_{j=1}^{|\Phi|} \lambda_j T_{ji}}. \qquad (4.4)$$

У даній формулі, оскільки повинна виконуватися умова для вибраного *i*-го типу МКП (або тактової частоти)

$$\forall j \in |\Phi| \;\; T_{ji} \le t_j^{\text{гр}}, \qquad (4.5)$$

то за (4.4) можна провести вибір для проекту такого типу МКП мінімальної продуктивності (і вартості), при якому виконується (4.5). Метод обчислення T_{ji} викладено в наступному пункті.

Знання розробником в процесі проектування системи керування реального часу значень $\forall j \in |\Phi| \;\; T_{ji}$ дозволяє для вибраного *i*-го типа мікропроцесора оцінити, яку частку завантаження процесора дає надання *j*-го сервісу (*j*-ї φ-транзакції)

$$\rho_{ji} = \lambda_j T_{ji} \quad \text{і} \quad \forall j \in |\Phi| \quad \rho_{ji} \le 0,6. \qquad (4.6)$$

Виконуючи розподіл оброблюваних φ-транзакцій по отриманих в результаті структурної оптимізації підсистемах (див. опис КСІ, п. 5.4), необхідно стежити, що б в проекті загальне завантаження сервісами кожної підсистеми $f_p \in \mathrm{F}$ не перевищувала допустимої межі (для процесорів це 0,6), тобто

$$\forall f_{pi} \in \mathrm{F}_{\mathbf{i}} \;\; R_{pi} \le 0,6. \qquad (4.7)$$

Іноді встановлюють і нижню межу завантаження (4.8) з метою ефективного використання обчислювальних ресурсів і зниження загальної вартості ІКОС.

$$\forall f_{pi} \in \mathrm{F}_{\mathbf{i}} \;\; 0,1 \le R_{pi} \le 0,6. \qquad (4.8)$$

Побудова φ-транзакцій дозволяє оцінити ряд інших дуже важливих

для проектування АСК характеристик, зокрема тих, що пов'язані з інформаційними потоками в майбутній системі. У АСК виділятимемо три типи інформаційних потоків: потоки повідомлень (msg), потоки керуючих дій (сигналів, s_k) і потоки на читання і запис інформації з/до масивів майбутньої бази даних.

Інформаційний потік повідомлень, що формується кожною φ-транзакцією j, і передається на периферійний пристрій d_k

$$\pi_{jd_k}^{msg} = \sum_{\forall \phi_l \in \varphi_j \wedge \exists d_k} \lambda_j \hat{p}_{jl} Q_{ld_k j} . \qquad (4.9)$$

Загальний сумарний потік повідомлень на приймач d_k в процесі роботи системи складатиме

$$\Pi_{d_k}^{msg} = \sum_{j=1}^{|\Phi|} \pi_{jd_k}^{msg} . \qquad (4.10)$$

У φ-транзакціях можуть формуватися потоки керуючих дій, або сигналів s_k, які виводяться через ПЗО на пристрої об'єкту керування. Для однієї j-ї транзакції і при кодуванні однієї дії 2-ма байтами (16 біт) вони складуть

$$\pi_{s_{kj}} = \sum_{\forall \phi_l \in \varphi_j \wedge \exists s_k} 16\lambda_j \hat{p}_{jl} . \qquad (4.11)$$

У системі в цілому потік сигналів s_k складатиме

$$\Pi_{s_k} = \sum_{j=1}^{|\Phi|} \pi_{s_{kj}} . \qquad (4.12)$$

Потоки інформації, пов'язані з читанням/записом даних з/до масиву i, при виконанні j-ї транзакції можна обчислити таким чином:

$$\pi_{ji}^{\text{чт/зп}} = \sum_{\forall \phi_l \in \varphi_j} \lambda_j \hat{p}_{jl} k_{j,l,i} O_i op_{j,l,i} . \qquad (4.13)$$

Сумарний інформаційний потік до масиву *i* в процесі роботи системи можна обчислити за формулою

$$\Pi_i^{\text{чт/зп}} = \sum_{j=1}^{|\Phi|} \pi_{ji}^{\text{чт/зп}} . \qquad (4.14)$$

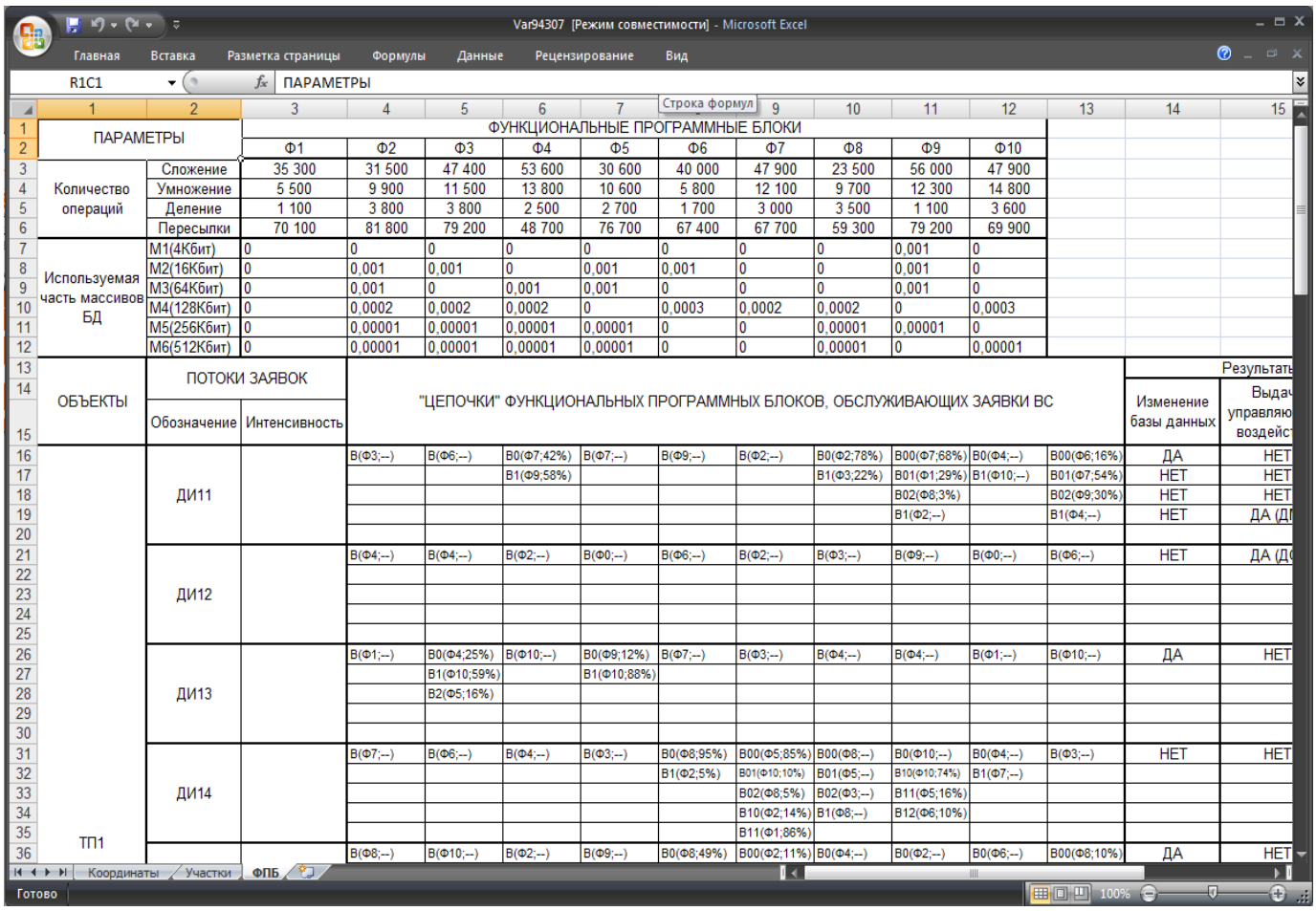
Загальний інформаційний потік до всіх масивів або потік до центральної бази даних АСК складатиме

$$\Pi_M^{\text{чт/зп}} = \sum_{i=1}^{|M|} \Pi_i^{\text{чт/зп}} . \qquad (4.15)$$

4.2 Метод обробки φ-транзакцій і розрахунку їх характеристик

φ-транзакції, визначені в п. 4.1, є фактично графами зі зваженими вершинами і ребрами (рис. 4.1). Для обробки таких графів існує велике різноманіття методів и методик, що залежать від поставлених завдань [121, 26]. Тут пропонується метод обробки, орієнтований на автоматизацію процесів перетворення φ-транзакцій і оцінку їх характеристик, що використовуються в процесі проектування ІКОС.

Розроблені графові і аналітичні засоби були покладені в основу програми CSProject [23] для автоматизованого формування транзакцій системи, які прив'язані до сигналів ділянок технологічних процесів сортувальних станцій. Початкові дані для розрахунків описані в п. 4.1. Вони можуть вводитись у програму двома способами: з попередньо підготовлених таблиць Excel спеціальної структури, фрагмент якої представлений на рисунку 4.3; в інтерактивному режимі з допомогою спеціально розробленого графічного редактора транзакцій і їх параметрів (див. рис. 4.4). Фрагмент результатів, отриманих з допомогою CSProject наведений на рис. 4.5.



Var94307 [Режим совместимости] - Microsoft Excel

Главная | Вставка | Разметка страницы | Формулы | Данные | Рецензирование | Вид

R1C1 | ПАРАМЕТРЫ

ПАРАМЕТРЫ		ФУНКЦИОНАЛЬНЫЕ ПРОГРАММНЫЕ БЛОКИ									
		Ф1	Ф2	Ф3	Ф4	Ф5	Ф6	Ф7	Ф8	Ф9	Ф10
Количество операций	Сложение	35 300	31 500	47 400	53 600	30 600	40 000	47 900	23 500	56 000	47 900
	Умножение	5 500	9 900	11 500	13 800	10 600	5 800	12 100	9 700	12 300	14 800
	Деление	1 100	3 800	3 800	2 500	2 700	1 700	3 000	3 500	1 100	3 600
	Пересылки	70 100	81 800	79 200	48 700	76 700	67 400	67 700	59 300	79 200	69 900
Используемая часть массивов БД	М1(4Кбит)	0	0	0	0	0	0	0	0	0,001	0
	М2(16Кбит)	0	0,001	0,001	0	0,001	0,001	0	0	0,001	0
	М3(64Кбит)	0	0,001	0	0,001	0,001	0	0	0	0,001	0
	М4(128Кбит)	0	0,0002	0,0002	0,0002	0	0,0003	0,0002	0,0002	0	0,0003
	М5(256Кбит)	0	0,00001	0,00001	0,00001	0,00001	0	0	0,00001	0,00001	0
	М6(512Кбит)	0	0,00001	0,00001	0,00001	0,00001	0	0	0,00001	0	0,00001

ОБЪЕКТЫ	ПОТОКИ ЗАЯВОК		"ЦЕПОЧКИ" ФУНКЦИОНАЛЬНЫХ ПРОГРАММНЫХ БЛОКОВ, ОБСЛУЖИВАЮЩИХ ЗАЯВКИ ВС									Результаты	
	Обозначение	Интенсивность										Изменение базы данных	Выдач управляю воздейст
ТП1	ДИ11		В(Ф3;--)	В(Ф6;--)	В0(Ф7;42%) В1(Ф9;58%)	В(Ф7;--)	В(Ф9;--)	В(Ф2;--)	В0(Ф2;78%) В1(Ф3;22%)	В00(Ф7;68%) В01(Ф1;29%) В02(Ф8;3%) В1(Ф2;--)	В0(Ф4;--) В1(Ф10;--)	В00(Ф6;16%) В01(Ф7;54%) В02(Ф9;30%) В1(Ф4;--)	ДА НЕТ НЕТ НЕТ	НЕТ НЕТ НЕТ ДА (ДІ
	ДИ12		В(Ф4;--)	В(Ф4;--)	В(Ф2;--)	В(Ф0;--)	В(Ф6;--)	В(Ф2;--)	В(Ф3;--)	В(Ф9;--)	В(Ф0;--)	В(Ф6;--)	НЕТ	ДА (Д
	ДИ13		В(Ф1;--)	В0(Ф4;25%) В1(Ф10;59%) В2(Ф5;16%)	В(Ф10;--)	В0(Ф9;12%) В1(Ф10;88%)	В(Ф7;--)	В(Ф3;--)	В(Ф4;--)	В(Ф4;--)	В(Ф1;--)	В(Ф10;--)	ДА	НЕТ
	ДИ14		В(Ф7;--)	В(Ф6;--)	В(Ф4;--)	В(Ф3;--)	В0(Ф8;95%) В1(Ф2;5%)	В00(Ф5;85%) В01(Ф10;10%) В02(Ф8;5%) В10(Ф2;14%) В11(Ф1;86%)	В00(Ф8;--) В01(Ф5;--) В02(Ф3;--) В1(Ф8;--)	В0(Ф10;--) В10(Ф10;74%) В11(Ф5;16%) В12(Ф6;10%)	В0(Ф4;--) В1(Ф7;--)	В(Ф3;--)	НЕТ	НЕТ
			В(Ф8;--)	В(Ф10;--)	В(Ф2;--)	В(Ф9;--)	В0(Ф8;49%)	В00(Ф2;11%)	В0(Ф4;--)	В0(Ф2;--)	В0(Ф6;--)	В00(Ф8;10%)	ДА	НЕТ

Координаты | Участки | ФПБ

Готово

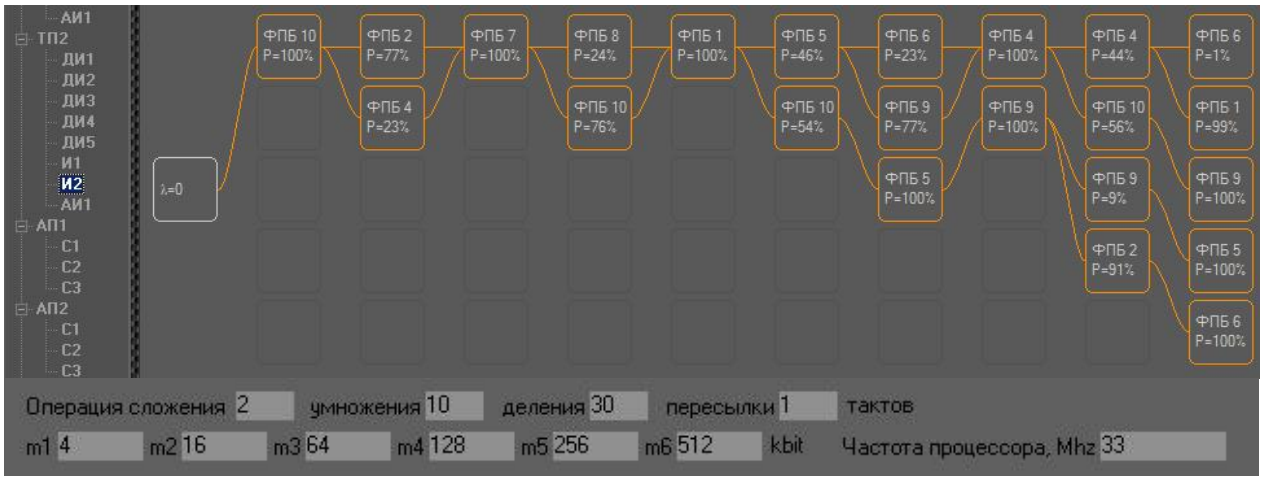


Рис. 4.3 Таблична формалізація початкових даних для проектування

Рис. 4.4 Вікно редактора вводу транзакції для сигналу И2 з технологічного процесу (ТП2) і вікно завдання параметрів масивів та МКК

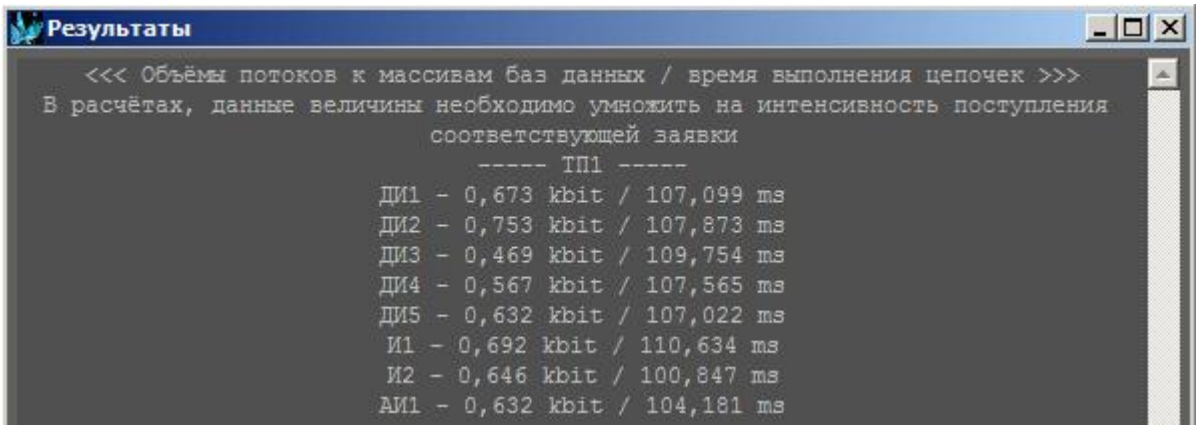


Рис. 4.5 Вікно результатів обробки φ-транзакції

4.3 Методика, аналітичні моделі та інструментальні засоби вибору системи пріоритетів обробки φ-транзакцій

Аналіз роботи сервіс-орієнтованих систем реального часу керування сортувальними станціями у вигляді набору φ-транзакцій, що обробляються, з жорсткими часовими обмеженнями показує, що в умовах попереднього вибраного типу мікропроцесора *i* повинні виконуватися безліч умов (4.25).

$$\forall j \in |\Phi| \ \forall d_k \in D \ \ t(\varphi_{ijd_k}) = t_{ijd_k} = T_{ijd_k}(\phi.\partial) + T_{ijd_k}(\phi.o) = \\ = \tau_{wijd_k} + T_{ijd_k} \le t_{ijd_k}^{\text{гр}}. \qquad (4.25)$$

Тобто для кожної транзакції j і кожної точки d_k виведення повідомлень або керуючих дій цією транзакцією, сума часів виконання фази диспетчерування (ф.д) і фази обробки транзакції (ф.о.) не повинно перевищувати встановлених граничних значень $t_{ijd_k}^{\text{гр}}$.

Оскільки T_{ijd_k} залежить тільки від вибраного типу мікропроцесора і, в даній постановці, є фіксованим, то умову (4.25) можна представити у вигляді

$$\forall j \in |\Phi| \ \forall d_k \in D \ \ \tau_{wijd_k} \le t_{ijd_k}^{\text{гр}} - T_{ijd_k}, \qquad (4.26)$$

τ_{wijd_k} можна варіювати, вибираючи дисципліну обслуговування заявок, що поступають в систему.

Тривалість інтервалів часу між надходженням заявок в ЦКС випадкова. У зв'язку з цим можуть виникнути черги заявок, які чекають обслуговування процесором. Організацію черг здійснюють спеціальні програми, які називаються «Диспетчерами». Схема обслуговування заявок в ЦКС показана на рисунку 4.6.

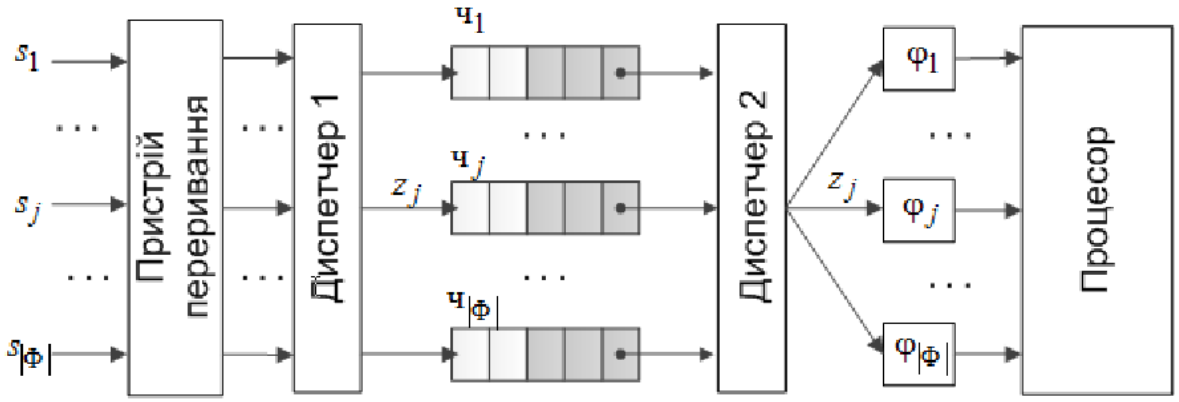


Рис. 4.6. Схема обслуговування заявок в ЦКС

Після надходження сигналу s_j уривається робота процесора, який можливо обслуговує заявку z_i, виконуючи транзакцію φ_i. Пристрій переривання ініціює роботу програми «Диспетчер 1», яка ставить заявку z_j в загальному випадку в одну з черг $ч_1,...,ч_{|\Phi|}$. Далі процесор продовжує перервану роботу. Після її завершення управління передається програмі «Диспетчер 2», яка здійснює аналіз стану черг для вибору заявки на обслуговування. Відповідно до коду вибраної заявки починається рішення φ-транзакції. Якщо черги порожні, то процесор знаходиться в стані очікування або вирішує так звані «фонові» задачі.

Правила, по яких здійснюється вибір заявок з черг, називаються дисципліною обслуговування. Якщо певним типам заявок не надається перевага (пріоритет), то дисципліну обслуговування називають безпріоритетною. В цьому випадку використовується одна черга заявок.

Вибір заявок z_j на обслуговування, як правило, виконується з початку черги або з її кінця. Першу дисципліну називають дисципліною FIFO (First Input - First Output), а другу - дисципліною LIFO (Last Input - First Output).

Названі дисципліни забезпечують однаковий середній час очікування заявок в черзі (τ_{wi}), проте дисципліна FIFO мінімізує дисперсію τ_{wi} і у зв'язку з цим застосовується найчастіше.

Характеристика розподілу ймовірності виникнення інтервалів між заявками визначається особливостями об'єкту керування. В даному випадку в задачах автоматизації сортувальних станцій приймаємо гіпотезу про експоненціальний характер цього розподілу [138].

Потік заявок з експоненціальним розподілом ймовірності називається простим, якщо виконуються умови незалежності надходження заявок і неможливості одночасного надходження декількох заявок. Прості потоки при об'єднанні утворюють також простий потік, причому їх інтенсивності λ_j підсумовуються. Для простого потоку характерне переважання коротких інтервалів між заявками. Тому створюється жорсткий режим роботи системи. Оцінки якості функціонування ІКОС, отримані в цьому випадку, є граничними.

Тривалість обслуговування τ_{si} з-за наявності різноманітних за трудомісткістю гілок програм також може бути випадковою. Її середнє значення визначається за методикою, викладеній в п. 4.1.2.

Коефіцієнт варіації ν_{si} величини τ_{si} може мінятися від 0 при постійному часі обслуговування до 1 при експоненціальному розподілі τ_{si}.

До важливих характеристик якості ЦКС відносять завантаження системи ρ_i, середній час очікування τ_{wi} і середній час перебування заявки в системі (час відповіді) τ_{sysi}. Величина завантаження системи *j*-м потоком заявок для *i*-го мікропроцесора $\rho_{ji} = \lambda_j \tau_{sji}$. Повне завантаження від всіх

потоків $R_{|\Phi|i} = \sum_{j=1}^{|\Phi|} \rho_{ji}$. Очевидно, що середній час відповіді можна знайти за формулою:

$$\tau_{sysji} = \tau_{sji} + \tau_{wji} \ . \qquad (4.27)$$

Для безпріоритетних дисциплін обслуговування (БП) середній час очікування для всіх типів заявок однаковий і може бути розрахований за формулою:

$$\tau_{wi} = \frac{\sum_{j=1}^{|\Phi|} \rho_{ji} \cdot \tau_{sji} \cdot (1 + \nu_{sji}^2)}{2 \cdot (1 - R_{|\Phi|i})}, \qquad (4.28)$$

де $|\Phi|$ – кількість типів заявок (кількість φ-транзакцій).

З формули (4.28) видно, що τ_{wi} збільшується у міру зростання ν_{sji}. Так, при експоненціальному законі розподілу τ_{sji} удвічі більше, ніж при постійному часі обслуговування. Це дозволяє спростити (4.28).

$$\tau_{wi} = \frac{\sum_{j=1}^{|\Phi|} \rho_{ji} \cdot \tau_{sji}}{(1 - R_{|\Phi|i})} . \qquad (4.29)$$

Відмінність терміновості рішення різних задач в ЦКС визначає доцільність надання заявкам певних типів переваги (пріоритету) перед іншими заявками (4.26). Зазвичай пріоритети характеризують цілими позитивними числами (1, 2, 3...).

Вважатимемо, що вищому пріоритету відповідає менше число. Пріоритети враховуються тільки у момент вибору заявки з черги, їх називають відносними (ВП). Хай кожному типу заявок відповідає свій відносний пріоритет. Тоді заявка z_j буде поставлена в чергу $ч_j$. Якщо при цьому процесор зайнятий виконанням заявки z_k (нижчого пріоритету $k > j$), то воно продовжуватиметься. Заявка z_j буде вибрана з черги по

закінченню виконання z_k, якщо до цього моменту не поступлять заявки з вищим пріоритетом.

Можливий варіант обслуговування, коли заявка, що поступила з вищим пріоритетом, захоплює процесор, обслуговуючий попередню заявку. В цьому випадку, говорять про абсолютний пріоритет (АП). Так, наприклад, при $j < k$ заявка z_j, яка поступила в чергу $ч_j$, буде прийнята на обслуговування. При цьому заявка, яка обслуговувалася z_k, повернеться в початок черги $ч_k$ і чекатиме обслуговування.

Очевидно, що привласнення заявкам відносних пріоритетів дозволяє зменшити τ_{wij} для заявок з високими пріоритетами за рахунок його збільшення для заявок з низькими пріоритетами. У разі застосування абсолютних пріоритетів цей ефект виявляється ще сильнішим.

Середній час очікування на *i*-м мікропроцесорі для заявок з відносним *j*-м пріоритетом:

$$\tau_{wij} = \frac{\sum_{j=1}^{|\Phi|} \rho_{ij} \cdot \tau_{sij} \cdot (1 + \nu_{sij}^2)}{2 \cdot (1 - R_{i(j-1)}) \cdot (1 - R_{ij})}. \qquad (4.30)$$

Формула для визначення середнього часу очікування на *i*-м мікропроцесорі при використанні *k*-го абсолютного пріоритету:

$$\tau_{wik} = \frac{\sum_{j=1}^{k} \rho_{ij} \cdot \tau_{sij} \cdot (1 + \nu_{sij}^2)}{2 \cdot (1 - R_{i(k-1)}) \cdot (1 - R_{ik})} + \frac{R_{i(k-1)} \cdot \tau_{sik}}{1 - R_{i(k-1)}}. \qquad (4.31)$$

Привласнення всім заявкам абсолютних або відносних пріоритетів може привести до надмірного збільшення τ_{wik} для заявок з низьким пріоритетом. В цьому випадку доцільне використання дисциплін обслуговування із змішаними пріоритетами (ЗП).

Значення τ_{wik} для системи із змішаними пріоритетами можна знайти за формулою (індекс *i*, що визначає тип мікропроцесора, для R, ρ, τ, ν в формулах опущені з метою спрощення запису):

$$\tau_{wk} = \begin{cases} \dfrac{R_{k-1} \cdot \tau_{sk}}{1 - R_{k-1}} + \dfrac{\sum_{j=1}^{k} \rho_j \cdot \tau_{sj} \cdot (1 + \nu_{sj}^2)}{2 \cdot (1 - R_{k-1}) \cdot (1 - R_k)} & (k = 1,...,M_1) - АП, \\ \dfrac{R_{M_1} \cdot \tau_{sk}}{1 - R_{M_1}} + \dfrac{\sum_{j=1}^{M} \rho_j \cdot \tau_{sj} \cdot (1 + \nu_{sj}^2)}{2 \cdot (1 - R_{k-1}) \cdot (1 - R_k)} & (k = M_1 + 1,...,M_1 + M_2) - ВП, \\ \dfrac{R_{M_1} \cdot \tau_{sk}}{1 - R_{M_1}} + \dfrac{\sum_{j=1}^{M} \rho_j \cdot \tau_{sj} \cdot (1 + \nu_{sj}^2)}{2 \cdot (1 - R_{M_1+M_2}) \cdot (1 - R_M)} & (k = M_1 + M_2 + 1,...,M_3) - БП \end{cases}, \quad (4.32)$$

де:

M_1 – кількість типів заявок з АП,

M_2 – кількість типів заявок з ВП,

M_3 – кількість типів заявок з БП.

Аналітичні вирази (4.28), (4.29), (4.30), (4.31) и (4.32) були отримані в роботах [121, 18]. Система позначень для моделей систем масового обслуговування (СМО) прийнята згідно роботи [101].

В даний час немає єдиної методики розподілу пріоритетів при дисципліні із змішаними пріоритетами [121]. У простому випадку це завдання може вирішуватися повним перебором всіх можливих варіантів розподілу пріоритетів і вибором найбільш прийнятного.

Для зменшення кількості переборів спочатку рекомендується переупорядкувати потоки відповідно до заданих обмежень на часи очікування, розташувати їх в порядку зростання часу обслуговування. Наприклад, якщо в систему поступає шість потоків заявок 1, 2 ., 6, допустимий час очікування заявок яких відповідно рівний (4.26):

$$\tau_{wi1}^{\text{гр}} = 1 \text{ с}, \ \tau_{wi2}^{\text{гр}} = 5 \text{ с}, \ \tau_{wi3}^{\text{гр}} = 2 \text{ с}, \ \tau_{wi4}^{\text{гр}} = 0{,}1 \text{ с}, \ \tau_{wi5}^{\text{гр}} = 0{,}5 \text{ с}, \ \tau_{wi6}^{\text{гр}} = 0{,}2 \text{ с},$$

то в результаті попереднього впорядковування потоків отримаємо таку послідовність 4, 6, 5, 1, 3, 2. Надалі пріоритети потокам заявок повинні призначатися таким чином: кожному потоку призначається пріоритет не нижче, ніж потокам, розташованим правіше, і не вище, ніж потокам, розташованим лівіше у вказаній послідовності. У даному прикладі потоку 4 повинен бути призначений пріоритет не нижче, ніж решті всіх потоків; потоку 6 -не нижче, ніж потокам 5, 1, 3, 2, і т.д. Потік 2 повинен мати пріоритет не вище, ніж решта всіх потоків.

Вигляд (абсолютний або відносний) і ступінь пріоритетності кожного потоку заявок по відношенню до інших потоків можуть вибиратися будь-яким способом. Одним з таких способів може бути цілеспрямований перебір деяких варіантів розподілу пріоритетів з урахуванням параметрів потоків заявок, обмежень на часи очікування заявок і ступеня зміни якості обслуговування при переході від одного варіанту до іншого. Цілеспрямованість полягає в тому, щоб кожен подальший варіант розподілу пріоритетів дозволяв задовольнити заданим обмеженням при вибраній швидкодії процесора.

Показником, що визначає необхідність зміни пріоритету деякого потоку, може служити відносне відхилення часу очікування τ_{wik}, отриманого при даному розподілі пріоритетів, від допустимого часу очікування $\tau_{wik}^{\text{гр}}$:

$$\delta_{wik} = \left|\tau_{wik}^{\text{гр}} - \tau_{wik}\right| / \tau_{wik} \ \ (k = 1,2,...,|\Phi|), \qquad (4.33)$$

де $|\Phi|$ — число потоків заявок, що поступають в систему.

Якщо відхилення для одних потоків значно відрізняються від відхилень для інших потоків, то необхідно змінити пріоритети цих потоків.

Зміна пріоритету потоку заявок може здійснюватися: 1) зміною пріоритету даного потоку по відношенню до інших потоків; 2) зміною пріоритетів інших потоків по відношенню до даного потоку. Розподіл пріоритетів задовільний, якщо для всіх потоків заявок значення δ_{wik},

приблизно однакові.

При проектуванні розподілених ІКОС завдання вибору пріоритетів ускладнюється, оскільки результат виконання φ-транзакції в підсистемі СМО 1 може передаватися в іншу підсистему СМО 5 через транзитні підсистеми СМО 3 по відповідних каналах передачі даних СМО 2 і СМО 4. Тобто необхідно розглядати не одну просту СМО, а мережу СМО (приклад такої мережі показаний на рис. 4.7).

Для розрахунку характеристик таких структур можна скористатися набором аналітичних моделей, описаних в [101], або використовувати імітаційне моделювання, наприклад, в середовищі JMT [171] (см. п. 5.4).

Декілька зауважень до рисунка 4.7.

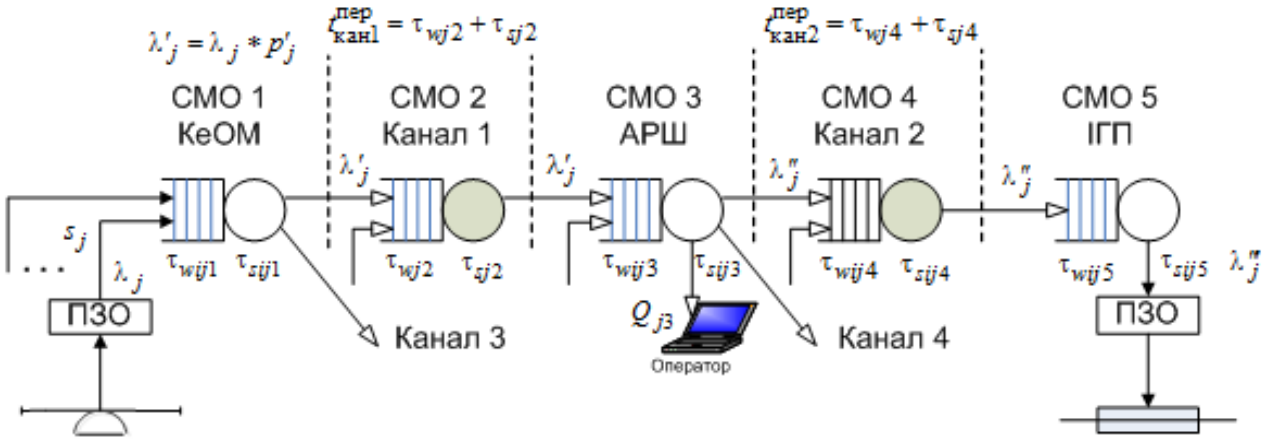


Рис. 4.7 Приклад мережі СМО обробки *j*-ї φ-транзакції

У кожній підсистемі обробки даних (СМО 1, СМО 3, СМО 5) інтенсивність вхідного потоку заявок може зменшуватися за рахунок розгалуження транзакції.

$$\lambda'_j = \lambda_j * p'_j;\ \ \lambda''_j = \lambda_j * p''_j;\ \ \lambda'''_j = \lambda_j * p'''_j,$$

де $0 < p_j^{\text{риски}} \le 1$.

Кожен канал (1 і 2) розглядається як СМО 2 і СМО 4, у якого час передачі визначається як $t_{\text{кан}}^{\text{пер}} = \tau_{wj(\text{кан})} + \tau_{sj(\text{кан})}$. Оскільки устаткування

каналів на ранніх стадіях ще не вибране, то така модель дозволяє оцінити вимоги до швидкості передачі даних в каналі. У даному прикладі тільки для одного *j*-го потоку

$$V_{j\text{кан}1} = \frac{Q_{j3}}{t_j^{\text{пер}}} = \frac{Q_{j3}}{t_{j3}^{\text{гр}} - \tau_{wij1} - \tau_{sij1}}. \qquad (4.34)$$

Очевидно, що сума всіх потоків дозволяє оцінити необхідну швидкість передачі даних для всіх видів інформаційних потоків (повідомлень і керуючих дій) всіх транзакцій, що використовують відповідний канал *k.*

$$V_k = \sum_{j \in K_k} V_j \,. \qquad (4.35)$$

Багатократне рішення задачі вибору пріоритетів в процесі проектування можна полегшити, використовуючи розроблену програму PRIORY.

4.4 Методи аналізу надійності систем з використанням теорії розмитих множин

4.4.1 Базові положення теорії для оцінки показників надійності

Виконані дослідження (розділ 3) свідчать про важливість задачі моделювання надійності при проектуванні систем [75]. Зазвичай надійна поведінка системи повністю описується імовірнісними характеристиками. Тим не менш, через неточності і невизначеності вихідних даних, оцінка точних значень ймовірностей в багатьох системах представляється складним завданням [178]. В останні роки деякі дослідники зосереджені на використанні теорії нечітких множин [214] для аналізу надійності нечітких системи [178, 183, 179. В [174] представлені наступні два припущення:

(1) Fuzzy-стан: в будь-який час система може перебувати або в стані

нечіткої працездатності або в стані нечіткої відмови.

(2) Припущення про можливість: поведінка системи може бути повністю охарактеризовано мірою можливості.

У роботі [178] представлено аналіз робіт, пов'язаних з введенням в інженерію систем з відмовами і використанням в ній нечіткої методології, аналізом надійності нечіткої системи з використанням нечітких числових арифметичних операцій, аналізом надійності нечіткої системи, заснований на нечітких часових рядах і α-відтинанні в операціях над нечіткими числами, нечітким аналізом надійності системи з використанням інтервалу довіри та інших відомих підходів і методів у даній області.

У цьому підпункті представляється новий табличний метод аналізу надійності нечіткої системи з використанням розмитих множин [183], [179], де надійнісні параметри компонентів системи видаються розмитими множинами, які визначаються в універсумі [0, 1]. Ступінь приналежності елемента *x* розмитій множині представляється розмитим значенням $[t_x, 1-f_x]$ в [0, 1], де t_x вказує на ступінь істини, f_x вказує на ступінь хибності, $1-t_x-f_x$ вказує на невідому частину $0 \le t_x \le 1-f_x \le 1$, і $t_x+f_x \le 1$. Поняття розмитих множин схоже на інтуїтивні нечіткі множини [169], і обидва вони є узагальненням поняття нечітких множин [214]. Пропонований метод табличного моделювання та аналізу надійності нечіткої системи володіє наочністю, гнучкістю і простою інтелектуальною формою.

Базові концепти розмитих множин. У 1965 році Заде запропонував теорію нечітких множин [214] або нечітка множина класів з нечіткими межами. Нехай маємо універсум U, що складається з n значень $U=\{u_1, u_2, ..., u_n\}$. Ступінь приналежності елемента u_i нечіткій множині представлені реальними значеннями між нулем і одиницею, де $u_i \in U$. В [183] Гау зазначив, що це одне значення ступеня приналежності поєднує в собі доказ, що $u_i \in U$, і демонструє протиріччя, що $u_i \in U$. Це не є

доказом, що $u_i \in U$, і доказом проти $u_i \in U$ відповідно, і воно не вказує, скільки там кожного з них. Крім того, Гау також зазначив, що одне значення ступеня нічого не говорить про його точності і представив в [183] основні поняття розмитих множини. В [179] запропоновані правила виконання арифметичних операцій над розмитими множинами. Нехай U універсум $U=\{u_1,u_2, ..., u_n\}$, який описується елементами. Розмита множина $\tilde{A}$ на універсумі U характеризується функцією приналежності істина (ФПІ) $t_{\tilde{A}}, t_{\tilde{A}} : U \to [0,1]$, і функцією приналежності хибності (ФПХ) $f_{\tilde{A}}$, $f_{\tilde{A}} : U \to [0,1]$, де $t_{\tilde{A}}(u_i)$ нижня межа функції приналежності u_i, що є доказом для значення u_i, $f_{\tilde{A}}(u_i)$ - нижня межа заперечення u_i, що є доказом проти значення u_i, і $t_{\tilde{A}}(u_i) + f_{\tilde{A}}(u_i) \le 1$. Значення функції приналежності u_i розмитій множині $\tilde{A}$ є обмежений підінтервал $[t_{\tilde{A}}(u_i), 1 - f_{\tilde{A}}(u_i)]$ в [0, 1]. Розмита величина $[t_{\tilde{A}}(u_i), 1 - f_{\tilde{A}}(u_i)]$ показує, що точне значення функції приналежності $\mu_{\tilde{A}}(u_i)$ для u_i обмежено діапазоном $t_{\tilde{A}}(u_i) \le \mu_{\tilde{A}}(u_i) \le 1 - f_{\tilde{A}}(u_i)$, де $t_{\tilde{A}}(u_i) + f_{\tilde{A}}(u_i) \le 1$. Приклад розмитої множини $\tilde{A}$ на універсумі U показано на рис. 4.8.

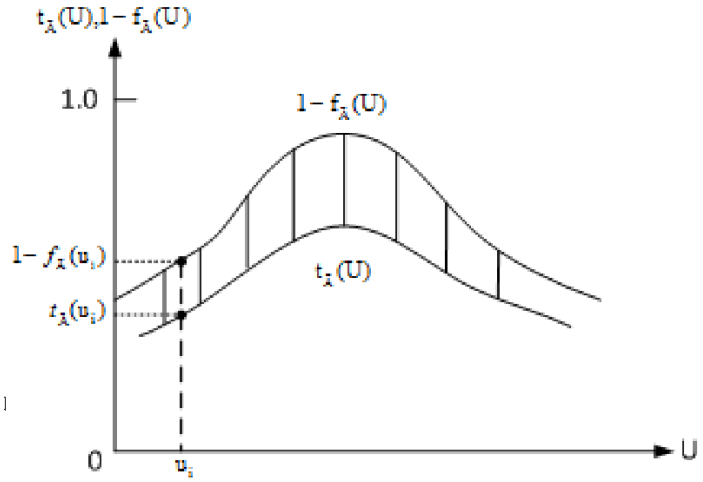


Рис. 4.8 Розмита множина

Якщо універс U є дискретною множиною, то розмита множина $\tilde{A}$ на універсі U може бути представлено як

$$\tilde{A} = \sum_{i=1}^{n} [t_{\tilde{A}}(u_i), 1 - f_{\tilde{A}}(u_i)] / u_i, \;\; u_i \in U. \qquad (4.36)$$

Якщо універс U є неперервною множиною, то розмита множина $\tilde{A}$ на універсі U може бути представлено як

$$\tilde{A} = \int_U [t_{\tilde{A}}(u_i), 1 - f_{\tilde{A}}(u_i)] / u_i, u_i \in U \qquad (4.37)$$

Визначення 1. Нехай $\tilde{A}$ розмита множина на універсі U з ФПІ $t_{\tilde{A}}$ і ФПХ $f_{\tilde{A}}$ відповідно. Розмита множина $\tilde{A}$ є випуклою тоді і лише тоді, коли для всіх u_1, u_2 на U,

$$t_{\tilde{A}}(\lambda u_1 + (1-\lambda)u_2) \geq \min(t_{\tilde{A}}(u_1), t_{\tilde{A}}(u_2)), \qquad (4.38)$$

$$1 - f_{\tilde{A}}(\lambda u_1 + (1-\lambda)u_2) \geq \min(1 - f_{\tilde{A}}(u_1), 1 - f_{\tilde{A}}(u_2)), \text{ где } \lambda \in [0,1]. \qquad (4.39)$$

Визначення 2. Розмита множина $\tilde{A}$ на універсі U називається нормальною розмитою множиною, якщо $\exists\, u_i \in U$ для котрого $1 - f_{\tilde{A}}(u_i) = 1$. В цьому випадку $f_{\tilde{A}}(u_i) = 0$.

Визначення 3. Розмите число є така розмита підмножина на універсі U, яка є як випуклою, так і нормальною.

Дальше, введемо основні арифметичні операції над трикутними розмитими множинами [179].

Розглянемо трикутну розмиту множину $\tilde{A}$, показану на рис. 4.9, де трикутна розмита множина $\tilde{A}$ може бути параметризована кортежем $<[(a,b,c);\mu_1], [(a,b,c);\mu_2]>$. Для зручності кортеж $<[(a,b,c);\mu_1], [(a,b,c);\mu_2]>$ будемо представляти абревіатурою $<[(a,b,c);\mu_1,\mu_2]>$, де $0 \leq \mu_1 \leq \mu_2 \leq 1$.

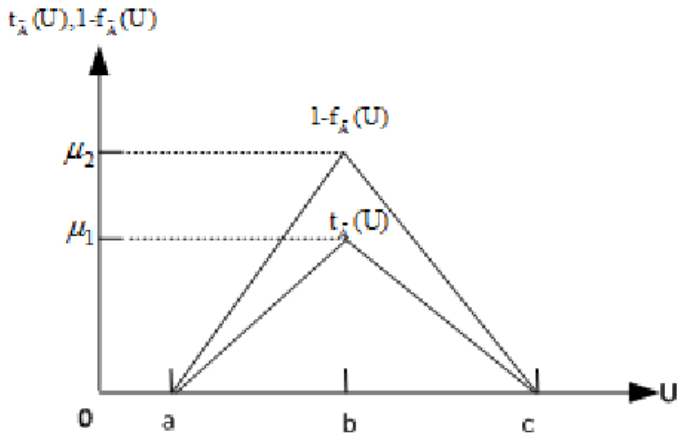


Рис. 4.9 Трикутна розмита множина

Тепер перейдемо до арифметичних операцій між трикутними розмитими множинами.

Випадок 1. Розглянемо трикутні розмиті множини $\tilde{A}$ і $\tilde{B}$, показані на рис. 4.10, де

$\tilde{A}$=<[(a_1,b_1,c_1);μ_1],[(a_1,b_1,c_1);μ_2]> = <[(a_1,b_1,c_1);μ_1;μ_2]>,

$\tilde{B}$=<[(a_2,b_2,c_2);μ_1],[(a_2,b_2,c_2);μ_2]> = <[(a_2,b_2,c_2);μ_1;μ_2]>,

і $0 \le \mu_1 \le \mu_2 \le 1$.

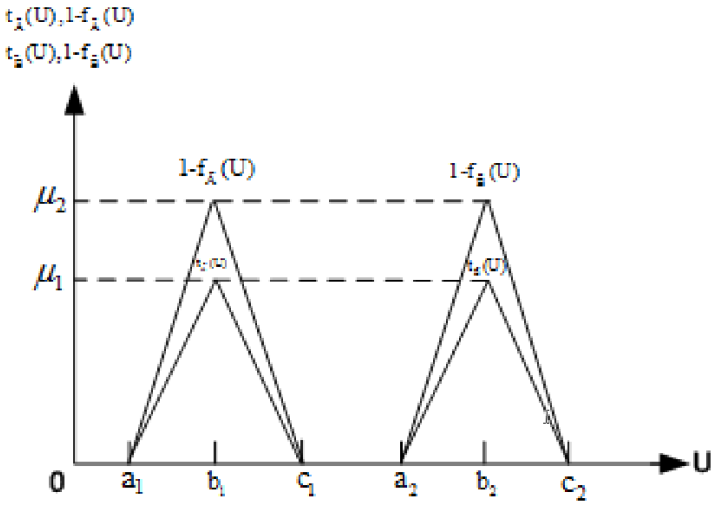


Рис. 4.10 Розмиті трикутні множини $\tilde{A}$ і $\tilde{B}$ (випадок 1)

Арифметичні операції між трикутними розмитими множинами $\tilde{A}$ і $\tilde{B}$ визначаються наступним чином.

Додавання ($\oplus$):

$$\begin{aligned}\tilde{A}\oplus\tilde{B}&=<[(a_1,b_1,c_1);\mu_1],[(a_1,b_1,c_1);\mu_2]>\oplus\\&\oplus<[(a_2,b_2,c_2);\mu_1],[(a_2,b_2,c_2);\mu_2]>=\\&=<[(a_1+a_2,b_1+b_2,c_1+c_2);\mu_1],[(a_1+a_2,b_1+b_2,c_1+c_2);\mu_2]>\\&=<[(a_1+a_2,b_1+b_2,c_1+c_2);\mu_1;\mu_2]>.\end{aligned}\qquad(4.40)$$

Віднімання ($!$):

$$\begin{aligned}\tilde{B}\,!\;\tilde{A}&=<[(a_2,b_2,c_2);\mu_1],[(a_2,b_2,c_2);\mu_2]>\,!\\&\;!\;<[(a_1,b_1,c_1);\mu_1],[(a_1,b_1,c_1);\mu_2]>=\\&=<[(a_2-c_1,b_2-b_1,c_2-a_1);\mu_1],[(a_2-c_1,b_2-b_1,c_2-a_1);\mu_2]>=\\&=<[(a_2-c_1,b_2-b_1,c_2-a_1);\mu_1;\mu_2]>.\end{aligned}\qquad(4.41)$$

Множення ($\otimes$):

$$\begin{aligned}\tilde{A}\otimes\tilde{B}&=<[(a_1,b_1,c_1);\mu_1],[(a_1,b_1,c_1);\mu_2]>\otimes\\&\otimes<[(a_2,b_2,c_2);\mu_1],[(a_2,b_2,c_2);\mu_2]>=\\&=<[(a_1\times a_2,b_1\times b_2,c_1\times c_2);\mu_1],[(a_1\times a_2,b_1\times b_2,c_1\times c_2);\mu_2]>=\\&=<[(a_1\times a_2,b_1\times b_2,c_1\times c_2);\mu_1;\mu_2]>.\end{aligned}\qquad(4.42)$$

Ділення ($\%$):

$$\begin{aligned}\tilde{B}\,\%\,\tilde{A}&=<[(a_2,b_2,c_2);\mu_1],[(a_2,b_2,c_2);\mu_2]>\,\%\\&\;\%<[(a_1,b_1,c_1);\mu_1],[(a_1,b_1,c_1);\mu_2]>=\\&=<[(a_2/c_1,b_2/b_1,c_2/a_1);\mu_1],[(a_2/c_1,b_2/b_1,c_2/a_1);\mu_2]>=\\&=<[(a_2/c_1,b_2/b_1,c_2/a_1);\mu_1;\mu_2]>.\end{aligned}\qquad(4.43)$$

Випадок 2. Розглянемо трикутні розмиті множини $\tilde{A}$ і $\tilde{B}$, показані на рис. 4.11, де

$$\tilde{A}=<[(a_1,b_1,c_1);\mu_1],[(a_1,b_1,c_1);\mu_2]>,$$

$$\tilde{B}=<[(a_2,b_2,c_2);\mu_3],[(a_2,b_2,c_2);\mu_4]>,$$

$$\text{і } 0\le\mu_3\le\mu_1\le\mu_4\le\mu_2\le 1.$$

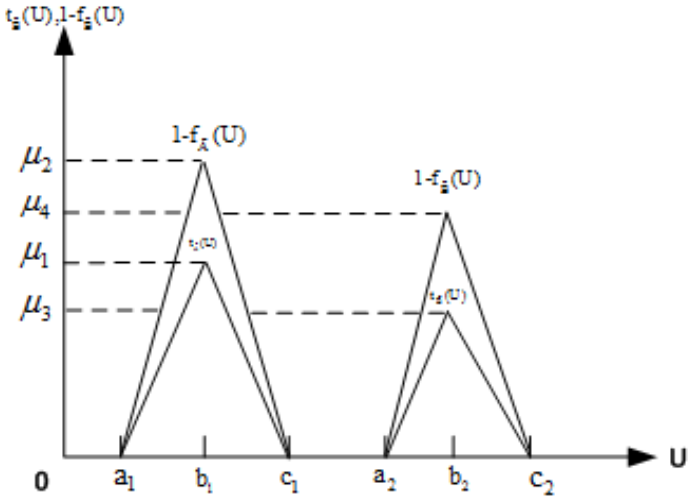


Рис. 4.11 Розмиті трикутні множини $\tilde{A}$ і $\tilde{B}$ (випадок 2)

Арифметичні операції в цьому випадку між трикутними розмитими множинами $\tilde{A}$ і $\tilde{B}$ визначаються наступним чином.

Додавання:

$$\begin{aligned}\tilde{A}\oplus\tilde{B}=&<[(a_1,b_1,c_1);\mu_1],[(a_1,b_1,c_1);\mu_2]>\oplus\otimes\\ &\oplus<[(a_2,b_2,c_2);\mu_3],[(a_2,b_2,c_2);\mu_4]>=\\ &=<[(a_1+a_2,b_1+b_2,c_1+c_2);\min(\mu_1,\mu_3)],\\ &\quad[(a_1+a_2,b_1+b_2,c_1+c_2);\min(\mu_2,\mu_4)]>.\end{aligned}\tag{4.44}$$

Віднімання:

$$\begin{aligned}\tilde{B}!\ \tilde{A}=&<[(a_2,b_2,c_2);\mu_3],[(a_2,b_2,c_2);\mu_4]>!\\ &!\ <[(a_1,b_1,c_1);\mu_1],[(a_1,b_1,c_1);\mu_2]>=\\ &=<[(a_2-c_1,b_2-b_1,c_2-a_1);\min(\mu_1-\mu_3)],\\ &\quad[(a_2-c_1,b_2-b_1,c_2-a_1);\min(\mu_2-\mu_4)]>.\end{aligned}\tag{4.45}$$

Множення:

$$\begin{aligned}\tilde{A}\otimes\tilde{B}=&<[(a_1,b_1,c_1);\mu_1],[(a_1,b_1,c_1);\mu_2]>\otimes\\ &\otimes<[(a_2,b_2,c_2);\mu_3],[(a_2,b_2,c_2);\mu_4]>=\\ &=<[(a_1\times a_2,b_1\times b_2,c_1\times c_2);\min(\mu_1,\mu_3)],\\ &\quad[(a_1\times a_2,b_1\times b_2,c_1\times c_2);\min(\mu_2,\mu_4)]>.\end{aligned}\tag{4.46}$$

Ділення:

$$\begin{aligned}\tilde{B}\%\tilde{A}=&<[(a_2,b_2,c_2);\mu_3],[(a_2,b_2,c_2);\mu_4]>\%\\ &\%<[(a_1,b_1,c_1);\mu_1],[(a_1,b_1,c_1);\mu_2]>=\\ &=<[(a_2/c_1,b_2/b_1,c_2/a_1);\min(\mu_1,\mu_3)],\\ &\quad[(a_2/c_1,b_2/b_1,c_2/a_1);\min(\mu_2,\mu_4)]>.\end{aligned}\tag{4.47}$$

4.4.2 Аналіз надійності нечітких систем на основі розмитих множин

Тут розглянемо новий метод аналізу надійності нечітких систем на основі теорії розмитих множин, де надійність компонентів системи представляються розмитими множинами, визначеними на універсі [0, 1].

Послідовні структури. Почнемо з послідовно надійнісної схеми з'єднань компонентів системи, як показано на рис. 4.12, де надійність $\tilde{R}_i$ компоненти P_i представляється розмитою множиною $<[(a_i,b_i,c_i);\ \mu_{i1};\mu_{i2}]>$,

де $0 \le \mu_{i1} \le \mu_{i2} \le 1$, а $1 \le i \le n$. Тоді надійність $\tilde{R}$ послідовної системи, показаної на рис. 4.12, може бути розрахована за формулою

$$\tilde{R}=\tilde{R}_1 \otimes \tilde{R}_2 \otimes ... \otimes \tilde{R}_n =$$

$$=<[(a_1,b_1,c_1); \mu_{11};\mu_{12}]>\otimes<[(a_2,b_2,c_2); \mu_{21};\mu_{22}]> \otimes ...$$
$$...\otimes<[(a_n,b_n,c_n); \mu_{n1};\mu_{n2}]> = \qquad (4.48)$$
$$=<[(\overset{n}{\underset{i=1}{\otimes}} a_i, \overset{n}{\underset{i=1}{\otimes}} b_i, \overset{n}{\underset{i=1}{\otimes}} c_i); \min(\mu_{11},\mu_{21}, ...,\mu_{n1}); \min(\mu_{12},\mu_{22}, ...,\mu_{n2})]>.$$

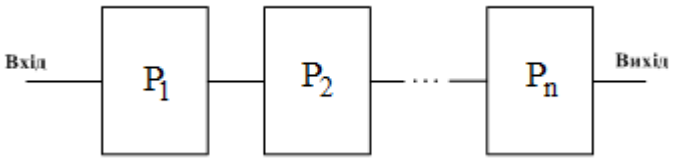


Рис. 4.12 Структура послідовної системи

Приклад 1.

Розглянемо приклад ланцюга з 3-х послідовно з'єднаних елементів. Вихідні значення параметрів і результат наведені на рис. 4.13.

	A	B	C	D	E	F	G	H	I	J	K	L	M
1													
2								ПРОИЗВЕД(C4:E4)					
3			R1	R2	R2	R	m_R						
4		a1	0,85	0,87	0,7	0,51765	0						
5		b1	0,9	0,95	0,92	0,7866	0,5						
6		c1	0,95	0,97	0,98	0,90307	0						
7		a2	0,85	0,87	0,7	0,51765	0						
8		b2	0,9	0,95	0,92	0,7866	0,3						
9		c2	0,95	0,97	0,98	0,90307	0						
10		m_1_max	0,5	0,6	0,7	0,5							
11		m_2_min	0,4	0,3	0,25	0,25							
12								МИН(C10:E10)					

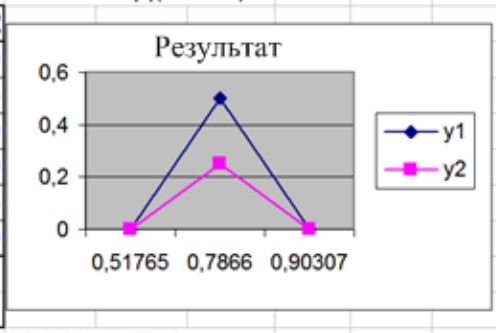


Рис. 4.13 Характеристики послідовної структури

Паралельні структури (системи з резервуванням). Тепер розглянемо паралельну систему, зображену на рис. 4.14, де надійність $\tilde{R}_i$ компоненти P_i представляється розмитою множиною $<[(a_i,b_i,c_i);\mu_{i1};\mu_{i2}]>$, де

$0 \le \mu_{i1} \le \mu_{i2} \le 1$, а $1 \le i \le n$. Тоді надійність $\tilde{R}$ паралельної системи, показаної на рис. 4.13, може бути розрахована за формулою

$$\tilde{R}=1! \bigotimes_{i=1}^{n}(1! \ \tilde{R}_i) = \tag{4.49}$$

$$=1! \ (1! \ <[(a_1,b_1,c_1);\mu_{11};\mu_{12}]>) \otimes (1! \ <[(a_2,b_2,c_2);\mu_{21};\mu_{22}]>) \otimes \ldots$$
$$\ldots \otimes (1! \ <[(a_n,b_n,c_n);\mu_{n1};\mu_{n2}]>) =$$
$$=1! \ <[(1-c_1,1-b_1,1-a_1);\mu_{11};\mu_{12}]> \otimes <[(1-c_2,1-b_2,1-a_2);\mu_{21};\mu_{22}]> \otimes \ldots$$
$$\ldots \otimes <[(1-c_n,1-b_n,1-a_n);\mu_{n1};\mu_{n2}]> =1! \ <(\bigotimes_{i=1}^{n}(1-c_i), \bigotimes_{i=1}^{n}(1-b_i), \bigotimes_{i=1}^{n}(1-a_i));$$
$$\min(\mu_{11};\mu_{21}; \ldots,\mu_{n1}); \min(\mu_{12};\mu_{22}; \ldots,\mu_{n2})]>=$$
$$=<[(1-\bigotimes_{i=1}^{n}(1-a_i),1-\bigotimes_{i=1}^{n}(1-b_i),1-\bigotimes_{i=1}^{n}(1-c_i));$$
$$\min(\mu_{11};\mu_{21}; \ldots,\mu_{n1}); \min(\mu_{12};\mu_{22}; \ldots,\mu_{n2})]>.$$

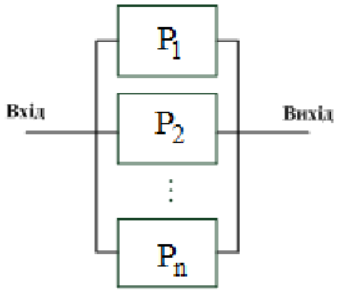


Рис. 4.14 Структура паралельної системи

Приклад 2.

Розглянемо приклад ланцюга з 3-х паралельно з'єднаних елементів. Вихідні значення параметрів і результати наведені в таблиці на рис. 4.15.

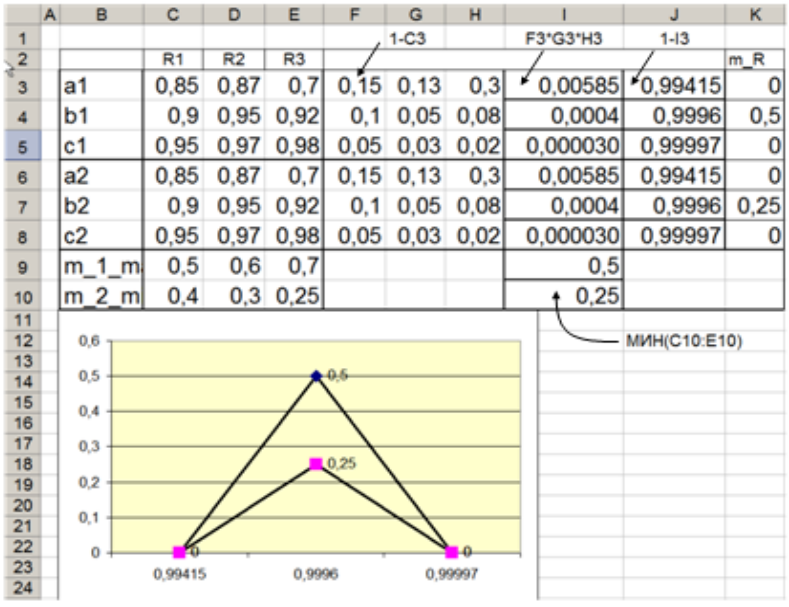


Рис. 4.15 Характеристики системи з потрійним резервуванням

Графічне представлення вихідних даних і результатів надійнісного моделювання наведені на рисунку 4.16.

Комбіновані структури. Розглянемо приклад, який ілюструє застосування описаного методу для аналізу надійності послідовно-паралельної структури нечіткої системи.

Приклад 3. Дослідимо систему, показану на рис.4.17, де надійність компонентів P_1, P_2, P_3 і P_4 є відповідно $\tilde{R}_1, \tilde{R}_2, \tilde{R}_3$ і $\tilde{R}_4$, де

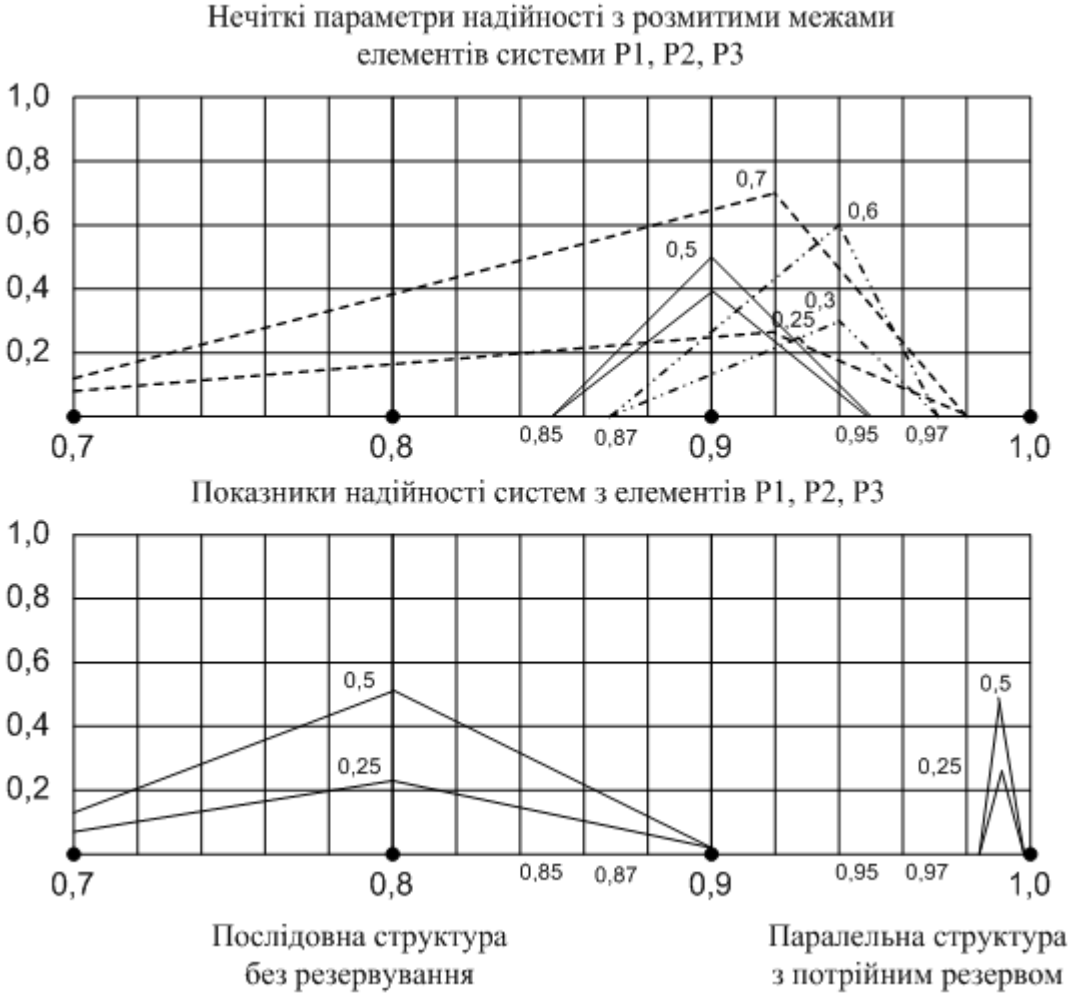


Рис. 4.16 Графічні результати рішення задач (рис. 4.14 і рис. 4.16)

$\tilde{R}_1 = <[(a_1,b_1,c_1);\mu_{11};\mu_{12}]>,$
$\tilde{R}_2 = <[(a_2,b_2,c_2);\mu_{21};\mu_{22}]>,$
$\tilde{R}_3 = <[(a_3,b_3,c_3);\mu_{31};\mu_{32}]>,$
$\tilde{R}_4 = <[(a_4,b_4,c_4);\mu_{41};\mu_{42}]>,$

$0 \le \mu_{i1} \le \mu_{i2} \le 1$ та $1 \le i \le 4$.

Базуючись на попередніх визначеннях, ми можемо оцінити надійність $\tilde{R}$ системи, наведеної на рис. 4.17, за допомогою наступних виразів:

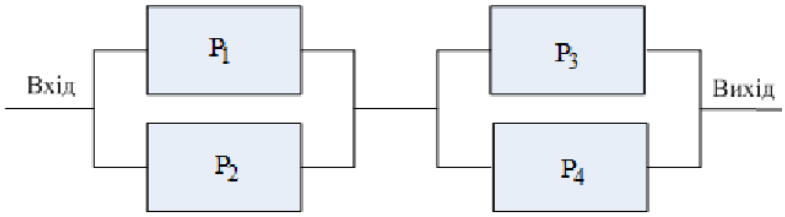


Рис. 4.17 Послідовно-паралельна структура

$$\tilde{R} = [1! \ (1! \ \tilde{R}_1) \otimes (1! \ \tilde{R}_2)] \otimes [1! \ (1! \ \tilde{R}_3) \otimes (1! \ \tilde{R}_4)] =$$

$$= [1! \ (1! \ <[(a_1,b_1,c_1);\mu_{11};\mu_{12}]>) \otimes (1! \ \lessdot[(a_2,b_2,c_2);\mu_{21};\mu_{22}]>) \otimes$$
$$\otimes [1! \ (1! \ <[(a_3,b_3,c_3);\mu_{31};\mu_{32}]>) \otimes (1! \ \lessdot[(a_4,b_4,c_4);\mu_{41};\mu_{42}]>)] =$$

$$= [1! \ <[(1-c_1,1-b_1,1-a_1);\mu_{11}; \ \mu_{12}]> \otimes <[(1-c_2,1-b_2,1-a_2);\mu_{21};\mu_{22}]>] \otimes$$
$$\otimes [1! \ <[(1-c_3,1-b_3,1-a_3);\mu_{31};\mu_{32}]> \otimes <[(1-c_4,1-b_4,1-a_4);\mu_{41};\mu_{42}]>] =$$

$$= [1! \ <[((1-c_1)(1-c_2),(1-b_1)(1-b_2),(1-a_1)(1-a_2));$$
$$\min(\mu_{11};\mu_{21}); \min(\mu_{12},\mu_{22})]>] \otimes$$
$$\otimes [1! \ <[((1-c_3)(1-c_4),(1-b_3)(1-b_4),(1-a_3)(1-a_4));$$
$$\min(\mu_{31};\mu_{41}); \min(\mu_{32},\mu_{42})]>] =$$

$$= <[1-(1-a_1)(1-a_2),1-(1-b_1)(1-b_2), \ 1-(1-c_1)(1-c_2));$$
$$\min(\mu_{11};\mu_{21});\min(\mu_{12},\mu_{22})]>] \otimes$$
$$\otimes <[(1-(1-a_3)(1-a_4),1-(1-b_3)(1-b_4),1-(1-c_3)(1-c_4));$$
$$\min(\mu_{31};\mu_{41}); \min(\mu_{32},\mu_{42})]> =$$

$$= <[(a_1 + a_2 - a_1a_2, \ b_1 + b_2 - b_1b_2, \ c_1 + c_2 - c_1c_2);$$
$$\min(\mu_{11},\mu_{21}); \ \min(\mu_{12},\mu_{22})] > \otimes$$
$$\otimes <[(a_3 + a_4 - a_3a_4, \ b_3 + b_4 - b_3b_4, \ c_3 + c_4 - c_3c_4);$$
$$\min(\mu_{31},\mu_{41}); \ \min(\mu_{32},\mu_{42})] > =$$

$$= <[((a_1+a_2 - a_1a_2)(a_3+a_4 - a_3a_4),(b_1+b_2-b_1b_2)(b_3+b_4 - b_3b_4),$$
$$(c_1+c_2 - c_1c_2)(c_3 \ +c_4 - c_3c_4));$$
$$\min(\mu_{11},\mu_{21},\mu_{31},\mu_{41}); \min(\mu_{12},\mu_{22},\mu_{32},\mu_{42})]> =$$

$$= <[(a_1a_3 + a_1a_4 - a_1a_3a_4 + a_2a_3 + a_2a_4 - \ a_2a_3a_4 - a_1a_2a_3 - a_1a_2a_4 + a_1a_2a_3a_4,$$
$$b_1b_3 + b_1b_4 - b_1b_3b_4 + b_2b_3 + b_2b_4 - b_2b_3b_4 - b_1b_2b_3 - b_1b_2b_4 + b_1b_2b_3b_4,$$
$$c_1c_3 + c_1c_4 - c_1c_3c_4 + c_2c_3 + c_2c_4 - c_2c_3c_4 - c_1c_2c_3 - c_1c_2c_4 + c_1c_2c_3c_4);$$
$$\min(\mu_{11},\mu_{21},\mu_{31},\mu_{41}); \min(\mu_{12},\mu_{22},\mu_{32},\mu_{42})]>.$$

Математичні вирази для оцінки надійності нечітких систем з розмитими параметрами отримати досить складно, особливо при побудові комбінованих структур. Більш ефективним і гнучким рішенням є

використання табличного опису взаємопов'язаних підсистем. Наприклад, структуру, зображену на рис. 4.17, можна представити у вигляді набору таблиць, наведених на рис. 4.14 і 4.16. В результаті отримуємо ефективну, персоналізовану модель для надійнісного моделювання нечітких систем з розмитими параметрами (рис. 4.18).

Запропонований метод табличного опису нечітких систем, елементи яких описуються трикутними функціями приналежності з розмитими межами, дозволяє моделювати характеристики надійності подібних систем шляхом побудови взаємопов'язаних таблиць для послідовних і паралельних структур без виконання складних математичних перетворень, що знижує ймовірність помилок, скорочує час побудови моделей і моделювання систем в цілому.

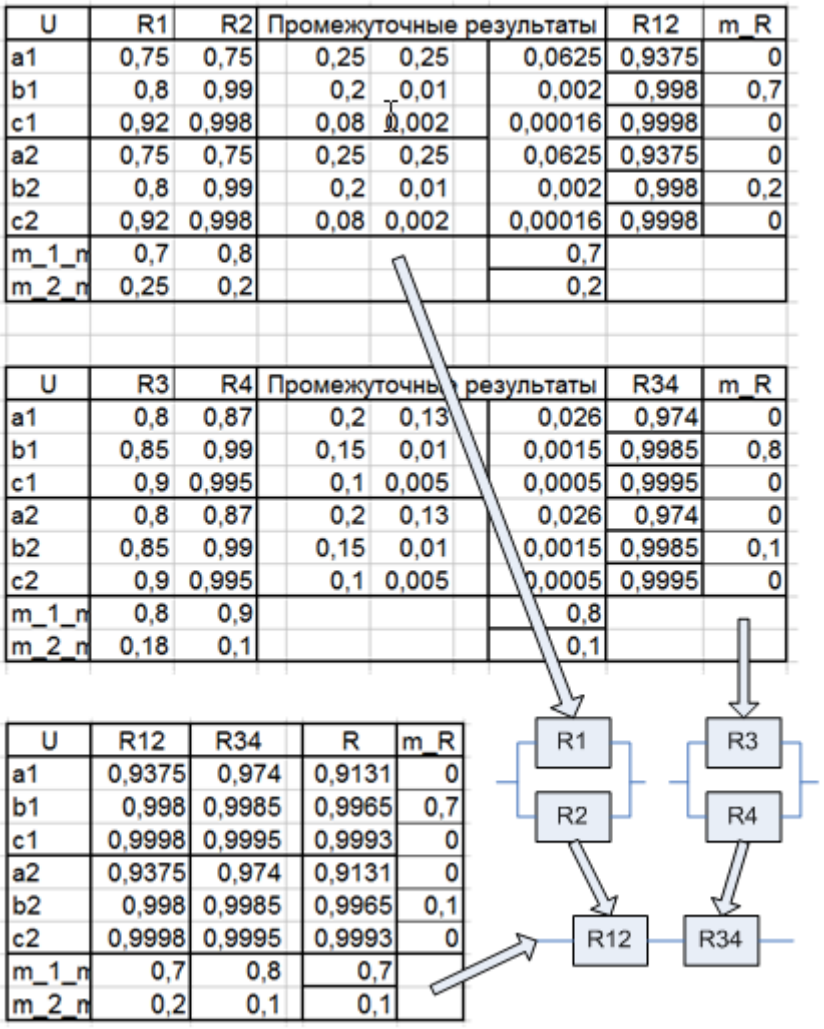


Рис. 4.18 Табличне моделювання нечітких систем

4.4.3 Метод використання різних видів розмитих множин для оцінки надійності нечітких систем

До цих пір в літературі обговорюються питання виконання арифметичних операцій між різними типами розмитих множин [75, 195, 214, 207, 174, 183, 156, 178]. Однак, як і раніше при аналізі надійності нечітких систем передбачається, що надійність всіх компонентів системи мають один і той же вид функцій приналежності [75]. Проте, в практичних завданнях така ситуація зустрічається рідко. Отже, необхіден метод, який дозволяв би знаходити надійність нечітких систем з компонентами, які мають різні типи функцій приналежності.

В методі пропонується уніфікований опис показників надійності нечітких систем з розмитими межами і різними видами функцій приналежності у вигляді кортежу $<[a_1, b_{11}, b_{12}, c_1]; [a_2, b_{21}, b_{22}, c_2]; \mu_1, \mu_2>$.

Для запропонованої структури кортежу розглядається новий алгоритм виконання різних арифметичних операцій між різними видами розпливчастих множин і пропонуються табличні методи аналізу надійності різних нечітких систем. Дослідження узагальнюють відомі роботи в цій області [75, 195, 156, 178, 183, 175].

У процесі проектування і технічного переозброєння сортувальних станцій, при використанні нових пристроїв, ми не можемо в чисельному вигляді висловити їх технічні характеристики і вплив різних чинників на процес функціонування сортувальної системи. Ці фактори зазвичай мають деяку невизначеність і мовні неточності в їх визначенні, що пов'язано з наступними обставинами.

Більшість систем управління на сортувальних станціях і гірках є занадто дорогими і складними, і експериментально виміряти їх характеристики в реальних умовах експлуатації досить складно, так як це пов'язане з виробничими втратами. Тому користуються експертними оцінками для виявлення збоїв в системах. Однак, оцінки експертів

зазвичай є невизначеними і ситуації описуються різними синтаксичними конструкціями.

Нормальний або ненормальний стан системи не можна точно визначити, тому що в ній реалізуються функції при деяких обмеженнях і не гарантується 100% надійність системи і її пристроїв.

Станційні системи автоматизації включають пристрої, побудовані на різній елементній базі – механічні, релейні, електронні. Ми не можемо виключити будь-яку можливість збоїв системи, включаючи системи електропостачання, природні умови і, звичайно, людський фактор.

Тому пропонується використовувати для аналізу надійності нечітких систем автоматизації сортувальних станцій розмиті множини, які допоможуть вирішити різні проблеми. У даному випадку експерти повинні тільки вказати діапазони рівнів довіри, що відповідають станам відмови в пристроях системи.

Визначення розмитих множин викладені в попередньому підпункті [183].

Розглянемо алгоритм виконання точних арифметичних операцій між різними типами розмитих множин $\tilde{A}_1, \tilde{A}_2, ..., \tilde{A}_n$ [195].

Для уніфікації описання різних трикутних і трапецієподібних функцій введемо наступний кортеж:

$$< [a_1, b_{11}, b_{12}, c_1]; [a_2, b_{21}, b_{22}, c_2]; \mu_1, \mu_2 >, \qquad (4.50)$$

де $a_1, b_{11}, b_{12}, c_1, a_2, b_{21}, b_{22}, c_2 \in U$ и $\mu_1, \mu_2 \in [0,1]$.

Він відповідає найбільш складному виду трапецієподібних функцій з незбіжними показниками надійності в кінцевих і екстремальних точках. У разі трикутних функцій $b_{11} = b_{12}$ і $b_{21} = b_{22}$. Можливі варіанти опису функцій цим кортежем представлені в прикладах і зведені в таблицю 5.3.

Крок 1

Знайти значення

$$\alpha_i = \sup_{i=0,1,...,n} (t_{\tilde{A}_i}(U)) = \max_{u_i \in U} \{t_{\tilde{A}}(u_i)\},$$

$$\beta_i = \sup_{i=0,1,...,n} (1 - f_{\tilde{A}_i}(U)) = \max_{u_i \in U} \{1 - f_{\tilde{A}}(u_i)\}.$$

Крок 2

Знайти $\forall i = 0,1,2,...,n$ значення $\alpha = \min(\alpha_i)$ і $\beta = \min(\beta_i)$.

Крок 3

Знайти інтервали певних значень $t_{\tilde{A}_i}(U)$, які належать діапазону $[0, \alpha]$. Нехай інтервал для $t_{\tilde{A}_i}(U) = p$, $0 \le p \le \alpha$ є $[u_{i1}, u_{i2}]$, де $u_{i1}, u_{i2} \in U$.

Крок 4

Знайти інтервали певних значень $1 - f_{\tilde{A}_i}(U)$, які належать діапазону $[0, \beta]$. Нехай інтервал для $1 - f_{\tilde{A}_i}(U) = q$, $0 \le q \le \beta$ є $[u'_{i1}, u'_{i2}]$, де $u'_{i1}, u'_{i2} \in U$.

Крок 5

Визначимо додавання, множення і віднімання розмитих множин для $t_{\tilde{A}}(U) = p$ як $[\sum_{i=1}^{n} u_{i1}, \sum_{i=1}^{n} u_{i2}]$, $[\prod_{i=1}^{n} u_{i1}, \prod_{i=1}^{n} u_{i2}]$ і $[u_{11} - (\sum_{i=2}^{n} u_{i2}), u_{12} - (\sum_{i=2}^{n} u_{i1})]$ відповідно, де $\tilde{A}$ представляє результат - розмиту множину.

Крок 6

Визначимо додавання, множення і віднімання розмитих множин для

$$1 - f_{\tilde{A}}(U) = q \text{ як } [\sum_{i=1}^{n} u'_{i1}, \sum_{i=1}^{n} u'_{i2}], [\prod_{i=1}^{n} u'_{i1}, \prod_{i=1}^{n} u'_{i2}] \text{ і}$$

$$[u'_{11} - (\sum_{i=2}^{n} u'_{i2}),\ u'_{12} - (\sum_{i=2}^{n} u'_{i1})]$$ відповідно.

Крок 7

Нарисувати функції приналежності результуючих розмитих множин після знаходження інтервалів для певних значень p (включаючи 0 і α) і q (включаючи 0 і β).

Введена надмірність дозволяє формалізувати процес обчислення результуючих показників надійності та побудову відповідних функцій приналежності за заданим значенням $t_{\tilde{A}}(u_i)$ і $1 - f_{\tilde{A}}(u_i)$, тобто μ_1 і μ_2.

Для лівої (зростаючої) частини функції

$$u_i = a_1 + \frac{t_{\tilde{A}}(u_i)(b_{11} - a_1)}{\alpha_i}, \tag{4.51}$$

$$u_i = a_1 + \frac{(1 - f_{\tilde{A}}(u_i))(b_{11} - a_1)}{\beta_i}, \tag{4.52}$$

Для правої (спадаючої) частини функції

$$u_i = c_2 - \frac{t_{\tilde{A}}(u_i)(c_2 - b_{22})}{\alpha_i}, \tag{4.53}$$

$$u_i = c_2 - \frac{(1 - f_{\tilde{A}}(u_i))(c_2 - b_{22})}{\beta_i}, \tag{4.54}$$

Розглянемо застосування розробленого опису розмитих нечітких множин та запропонованого алгоритму для аналізу надійності послідовно, паралельно, паралельно-послідовних і послідовно-паралельних нечітких систем, де надійнісні параметри компонентів системи представлені різними типами розмитих множин, що визначаються на універсі [0, 1].

Послідовна система. Розглянемо послідовну систему, яка складається з ‘n’ компонентів і показана на рис. 4.19.

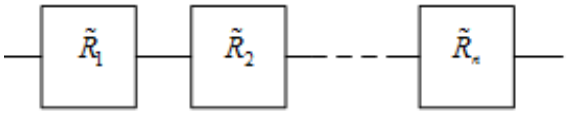


Рис. 4.19 Послідовна система

Нечітка надійність $\tilde{R}_S = \overset{n}{\underset{i=1}{\otimes}} \tilde{R}_i$ послідовної системи, показаної на рис. 4.19, може бути розрахована за алгоритмом, описаному в розділі 3 для множення, де $\tilde{R}_i$ представляє надійність *i*-ої компоненти.

Паралельна система. Розглянемо паралельну систему, яка складається з ‘n’ компонентів, показаних на рисунку 4.20.

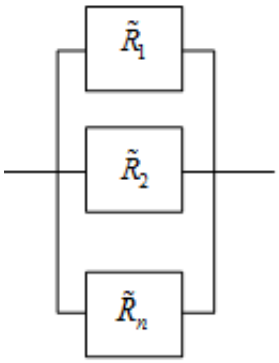


Рис. 4.20 Паралельна система

Нечітка надійність $\tilde{R}_P = 1! \overset{n}{\underset{i=1}{\otimes}} (1! \tilde{R}_i)$ паралельної системи, приведеної на рис. 4.20, може бути розрахована за алгоритмом, приведеному в секції 3, з використанням операцій множення і віднімання, де $\tilde{R}_i$ представляє надійність *i*-й компоненти.

Паралельно-послідовна система. Розглянемо паралельно-послідовну систему, яка складається з ‘m’ гілок, об’єднаних паралельно. В кожній гілці по ‘n’ компонентів, як показано на рис. 4.21.

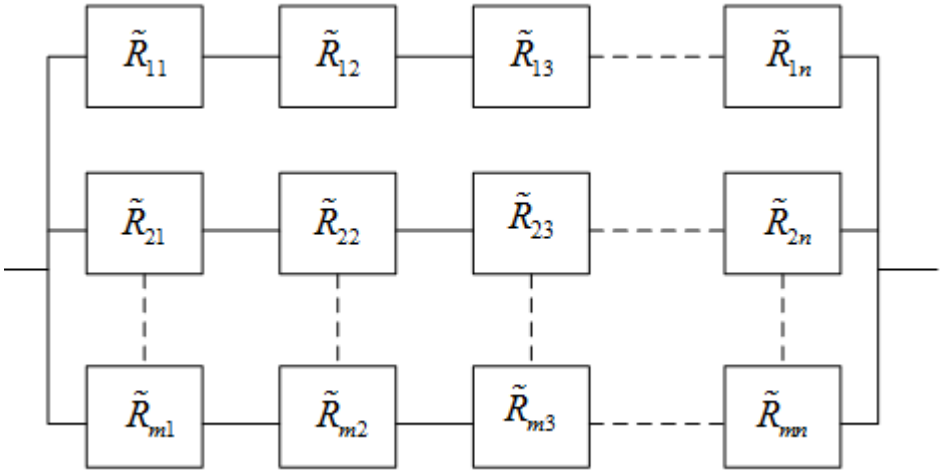


Рис. 4.21 Паралельно-послідовна система

Нечітка надійність $\tilde{R}_{PS} = 1! \underset{k=1}{\overset{m}{\otimes}} (1! \ (\underset{i=1}{\overset{n}{\otimes}} \tilde{R}_{ki}))$ паралельно-послідовної системи може бути розрахована за алгоритмом, наведеному в секції 3, за допомогою операцій множення і віднімання, де $\tilde{R}_{ki}$ представляє надійність i-ї компоненти (i=1,2,…,n) в k-ій гілці (k=1,2,…,m).

Послідовно-паралельна система. Розглянемо послідовно-паралельну систему, яка складається з ‘n’ послідовних підсистем, кожна з яких складається з ‘m’ паралельних компонентів (див. рис. 4.22).

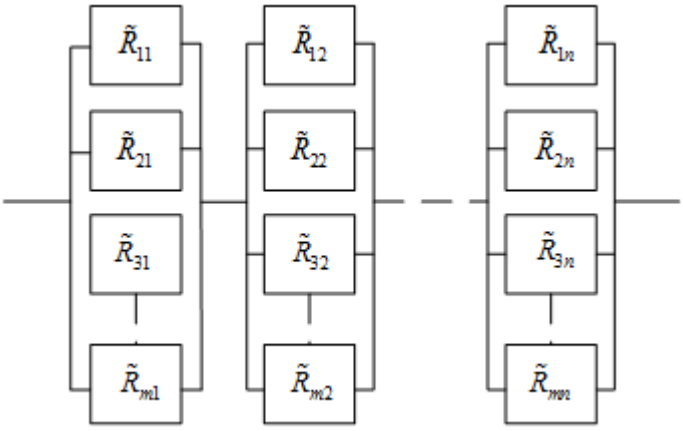


Рис. 4.22 Послідовно-паралельна система

Нечітка надійність $\tilde{R}_{SP} = \underset{k=1}{\overset{n}{\otimes}} (1! \ \underset{i=1}{\overset{m}{\otimes}} (1! \ \tilde{R}_{ik}))$ послідовно-паралельної системи може бути розрахована за алгоритмом, описаному в секції 3, за

допомогою операцій множення і віднімання, де $\tilde{R}_{ik}$ представляють надійність і-ої компоненти (i=1,2,…,m) в k-ій підсистемі (k=1,2,…,n).

4.4.4 Приклади рішення задач оцінки надійності нечітких систем без резервування і з багатократним резервом

Для ілюстрації розглянутого алгоритму розрахуємо нечітку надійність послідовної системи $\tilde{R}_S$ (рис. 4.23), паралельної системи $\tilde{R}_P$ (рис. 4.24), паралельно-послідовної системи $\tilde{R}_{PS}$ (рис. 4.27) і послідовно-паралельної системи $\tilde{R}_{SP}$ (рис. 4.26), кожна з яких складається з 4-х компонентів.

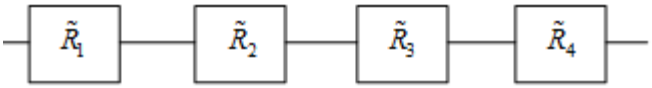


Рис. 4.23 Приклад послідовної системи

Послідовна схема відповідає, наприклад, ланцюгу підсистем в ієрархічній системі керування сортувальною станцією: НО-НА-МККпс-КеОМ.

Нехай надійність компонентів представлена різними типами розмитих множин $\tilde{R}_1, \tilde{R}_2, \tilde{R}_3$ і $\tilde{R}_4$, які визначаються на універсі R, де R представляються розмитими параметрами надійності на інтервалі [0,1].

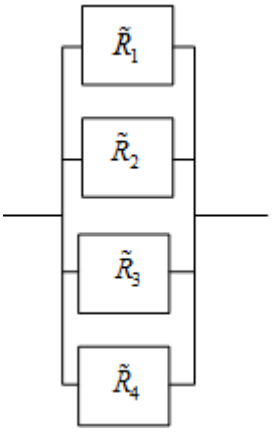


Рис. 4.24 Приклад паралельної системи з резервуванням

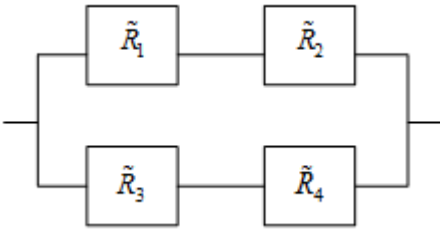


Рис. 4.25 Приклад паралельно-послідовної системи

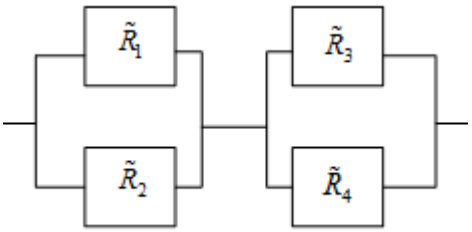


Рис. 4.26 Приклад послідовно-паралельної системи

Прийняті значення $\tilde{R}_1, \tilde{R}_2, \tilde{R}_3, \tilde{R}_4$ для обчислення $\tilde{R}_S, \tilde{R}_P, \tilde{R}_{PS}, \tilde{R}_{SP}$ зведені в таблицю 4.5 (з уніфікованим записом кортежів), а функції приналежності, які описують $\tilde{R}_1, \tilde{R}_2, \tilde{R}_3, \tilde{R}_4$, показані на рис. 4.27.

Таблиця 4.5

Нечітка надійність компонентів

Нечітка надійність $\tilde{R}_i$ і-ої компоненти	Тип розмитої множини
$\tilde{R}_1 = <[(0.2, 0.4, 0.4, 0.6); (0.2, 0.4, 0.4, 0.6); 0.7; 0.8]>$	Трикутна
$\tilde{R}_2 = <[(0.3, 0.5, 0.7, 0.9); (0.3, 0.5, 0.7, 0.9); 0.6; 0.9]>$	Трапецієподібна
$\tilde{R}_3 = <[(0.6, 0.7, 0.7, 0.8); (0.5, 0.7, 0.7, 0.9); 0.5; 0.7]>$	Трикутна
$\tilde{R}_4 = <[(0.2, 0.3, 0.4, 0.5); (0.1, 0.3, 0.4, 0.6); 0.4; 0.6]>$	Трапецієподібна
$\tilde{R}_S = \tilde{R}_1 \otimes \tilde{R}_2 \otimes \tilde{R}_3 \otimes \tilde{R}_4$	
$\tilde{R}_P = 1!\ (1!\ \tilde{R}_1) \otimes (1!\ \tilde{R}_2) \otimes (1!\ \tilde{R}_3) \otimes (1!\ \tilde{R}_4)$	
$\tilde{R}_{PS} = 1!\ (1!\ (\tilde{R}_1 \otimes \tilde{R}_2)) \otimes (1!\ (\tilde{R}_3 \otimes \tilde{R}_4))$	
$\tilde{R}_{SP} = 1!\ (1!\ \tilde{R}_1) \otimes (1!\ \tilde{R}_2)) \otimes (1!\ (1!\ \tilde{R}_3) \otimes (1!\ \tilde{R}_4))$	

Рішення прикладів

Крок 1

Знаходимо значення $\alpha_1, \alpha_2, \alpha_3, \alpha_4, \beta_1, \beta_2, \beta_3$ і β_4 з заданих значень $\tilde{R}_1, \tilde{R}_2, \tilde{R}_3$ і $\tilde{R}_4$:

$\alpha_1 = 0.7,\ \alpha_2 = 0.6,\ \alpha_3 = 0.5,\ \alpha_4 = 0.4;$

$\beta_1 = 0.8,\ \beta_2 = 0.9,\ \beta_3 = 0.7,\ \beta_4 = 0.6.$

Крок 2

Знаходимо

$\alpha = \min(0.7, 0.6, 0.5, 0.4) = 0.4$ і $\beta = \min(0.8, 0.9, 0.7, 0.6) = 0.6.$

Крок 3

Обчислюємо інтервали надійності для всіх компонентів, як точні значення $t_{\tilde{R}_i}(R)$, які належать діапазону [0,0.4] (див. таблицю 4.6).

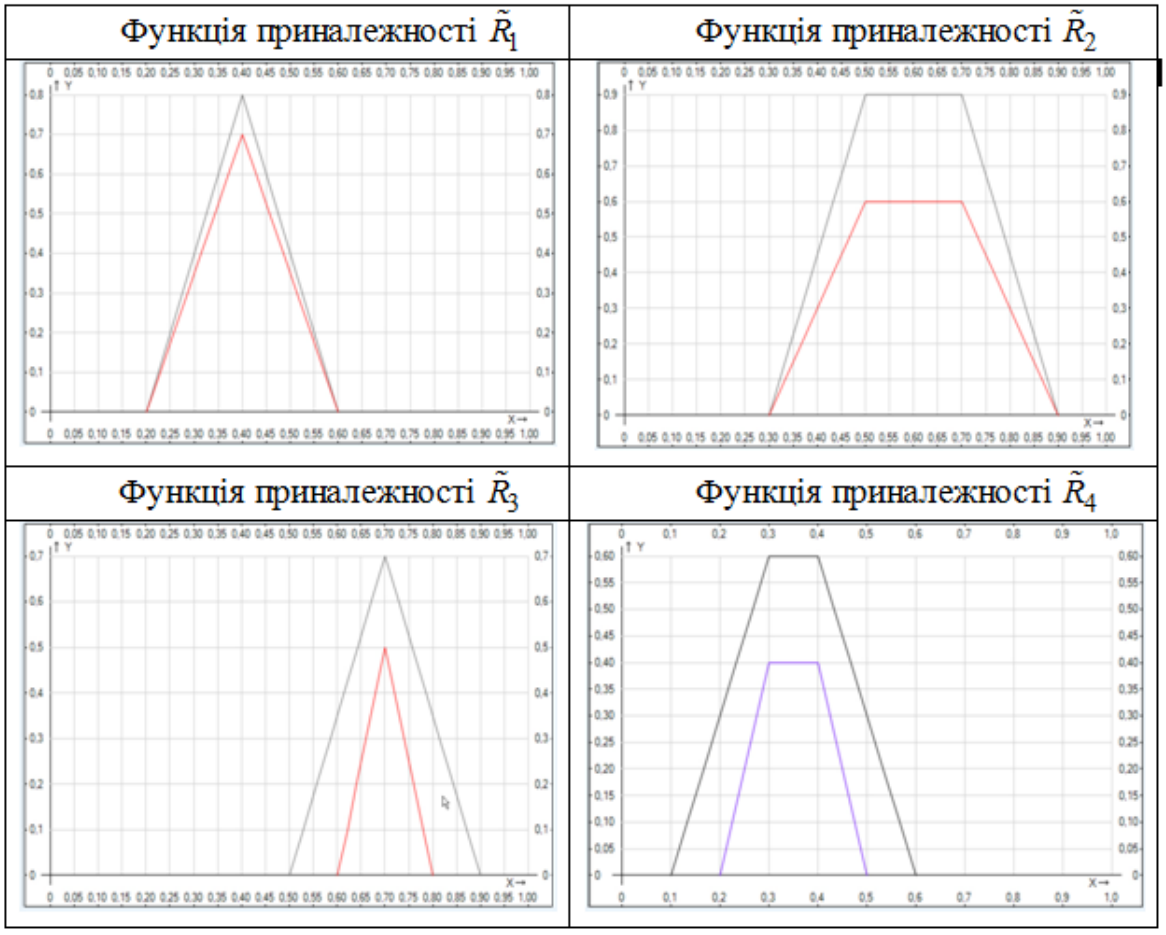


Рис. 4.27 Функції приналежності для компонентів з нечіткою надійністю

Таблиця 4.6

Інтервали надійності для точних значень $t_{\tilde{R}_i}(R)$

Значення $t_{\tilde{R}_i}(R)$	Інтервали надійності $\tilde{R}_1$	Інтервали надійності $\tilde{R}_2$	Інтервали надійності $\tilde{R}_3$	Інтервали надійності $\tilde{R}_4$
0	[0.200,0.600]	[0.300,0.900]	[0.600,0.800]	[0.200,0.500]
0.1	[0.229.0.571]	[0.333,0.867]	[0.620,0.780]	[0.225,0.475]
0.2	[0.257,0.543]	[0.367,0.833]	[0.640,0.760]	[0.250,0.450]
0.3	[0.286,0.514]	[0.400,0.800]	[0.660,0.740]	[0.275,0.425]
0.4	[0.314,0.486]	[0.433,0.767]	[0.680,0.720]	[0.300,0.400]

Крок 4

Обчислюємо інтервали надійності для всіх компонентів, як точні значення $1-f_{\tilde{R}_i}(R)$, які належать діапазону [0,0.6] (див. таблицю 4.7).

Таблиця 4.7

Інтервали надійності для точних значень $1-f_{\tilde{R}_i}(R)$

Значення $1-f_{\tilde{R}_i}(R)$	Інтервали надійності $\tilde{R}_1$	Інтервали надійності $\tilde{R}_2$	Інтервали надійності $\tilde{R}_3$	Інтервали надійності $\tilde{R}_4$
0	[0.200,0.600]	[0.300,0.900]	[0.500,0.900]	[0.100,0.600]
0.1	[0.225.0.575]	[0.322,0.878]	[0.529,0.871]	[0.133,0.567]
0.2	[0.250,0.550]	[0.344,0.856]	[0.557,0.843]	[0.167,0.533]
0.3	[0.275,0.525]	[0.367,0.833]	[0.586,0.814]	[0.200,0.500]
0.4	[0.300,0.500]	[0.389,0.811]	[0.614,0.786]	[0.233,0.467]
0.5	[0.325,0.475]	[0.411,0.789]	[0.643,0.757]	[0.267,0.433]
0.6	[0.350,0.450]	[0.433,0.767]	[0.671,0.729]	[0.300,0.400]

Отримані функції після виконання кроків 3 і 4 показані на рис. 4.28.

Крок 5

Інтервали надійності для послідовної, паралельної, паралельно-послідовної і послідовно-паралельної систем (див. рис. 4.23, 4.24, 4.25, 4.26) можна обчислити з використанням таблиць 4.5, 4.6, 4.7 і алгоритму в секції 3 для точних значень $t_{\tilde{R}_i}(R)$,які належать діапазону [0,0.4]. Отримані результати наведені в таблиці 4.8.

Крок 6

Інтервали надійності для послідовної, паралельної, паралельно-

послідовної і послідовно-паралельної систем (див. рис. 4.23, 4.24, 4.25, 4.26) можна обчислити з використанням таблиць 4.5, 4.6, 4.7 і алгоритму в секції 3 для точних значень $1-f_{\tilde{R}_i}(R)$, які належать діапазону [0,0.6]. Отримані результати наведені в таблиці 4.9.

Таблиця 4.8

Інтервали надійності для точних значень $t_{\tilde{R}_i}(R)$

Значення $t_{\tilde{R}_i}(R)$	Інтервали надійності $\tilde{R}_S$	Інтервали надійності $\tilde{R}_P$	Інтервали надійності $\tilde{R}_{PS}$	Інтервали надійності $\tilde{R}_{SP}$
0	[0.007,0.216]	[0.821,0.996]	[0.173,0.724]	[0.299,0.864]
0.1	[0.011.0.183]	[0.849,0.993]	[0.205,0.682]	[0.343,0.834]
0.2	[0.015,0.155]	[0.873,0.990]	[0.239,0.640]	[0.387,0.802]
0.3	[0.021,0.129]	[0.894,0.985]	[0.275,0.596]	[0.431,0.768]
0.4	[0.028,0.107]	[0.913,0.980]	[0.312,0.553]	[0.474,0.732]

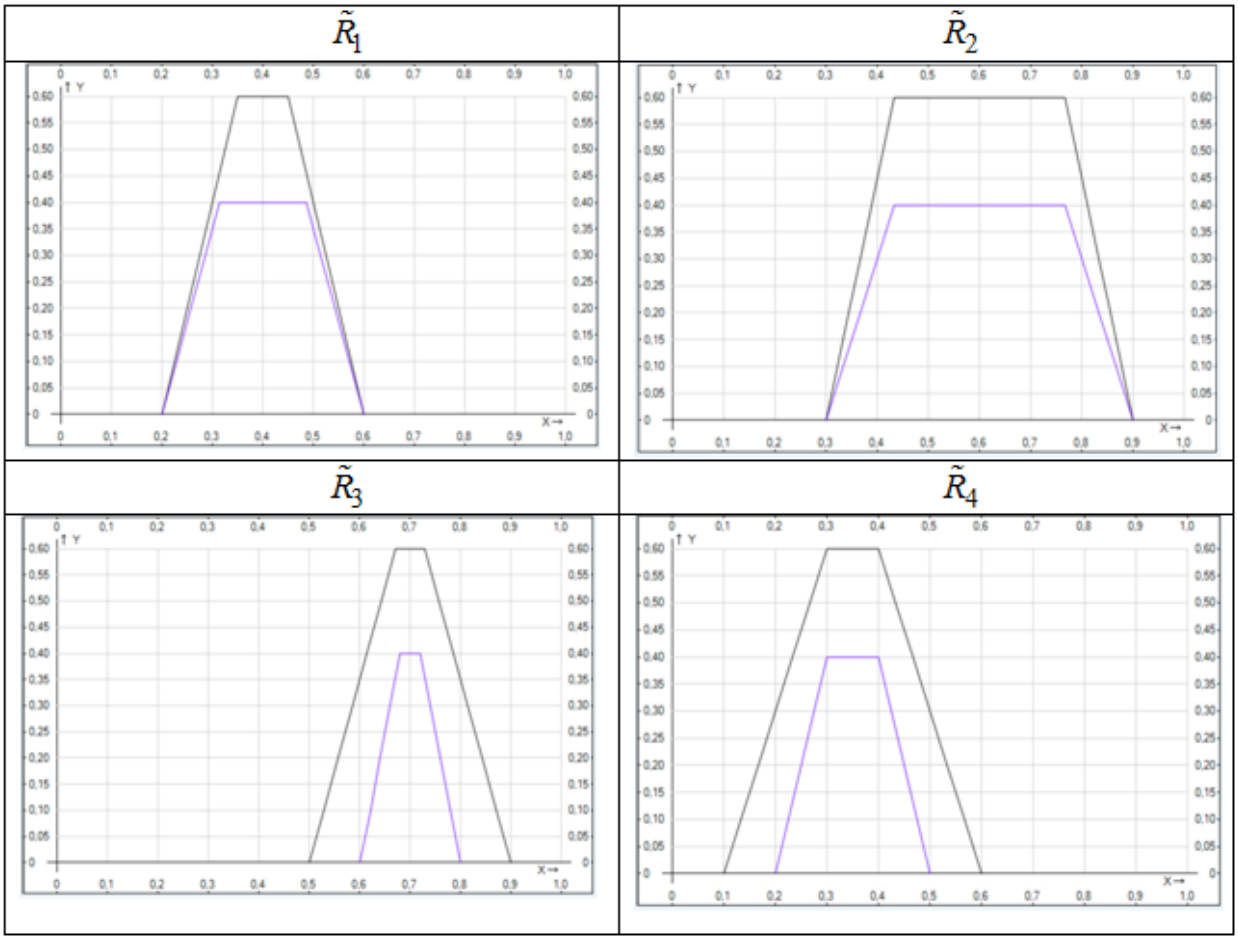


Рис. 4.28 Графіки функцій з α і β відтинанням

Крок 7

Функції приналежності, які представляються нечітку надійність послідовної, паралельної, паралельно-послідовної і послідовно-паралельної системи показані на рис. 4.29.

Таблиця 4.9

Інтервали надійності для точних значень $1-f_{\tilde{R}_i}(R)$

Значення $1-f_{\tilde{R}_i}(R)$	Інтервали надійності $\tilde{R}_S$	Інтервали надійності $\tilde{R}_P$	Інтервали надійності $\tilde{R}_{PS}$	Інтервали надійності $\tilde{R}_{SP}$
0	[0.003,0.292]	[0.748,0.998]	[0.107,0.788]	[0.242,0.922]
0.1	[0.005.0.249]	[0.785,0.997]	[0.138,0.749]	[0.281,0.895]
0.2	[0.008,0.212]	[0.818,0.995]	[0.171,0.709]	[0.321,0.867]
0.3	[0.012,0.178]	[0.848,0.993]	[0.206,0.666]	[0.362,0.835]
0.4	[0.017,0.149]	[0.873,0.989]	[0.243,0.624]	[0.403,0.802]
0.5	[0.023,0.123]	[0.896,0.985]	[0.282,0.580]	[0.445,0.767]
0.6	[0.031,0.101]	[0.915,0.979]	[0.322,0.536]	[0.486,0.730]

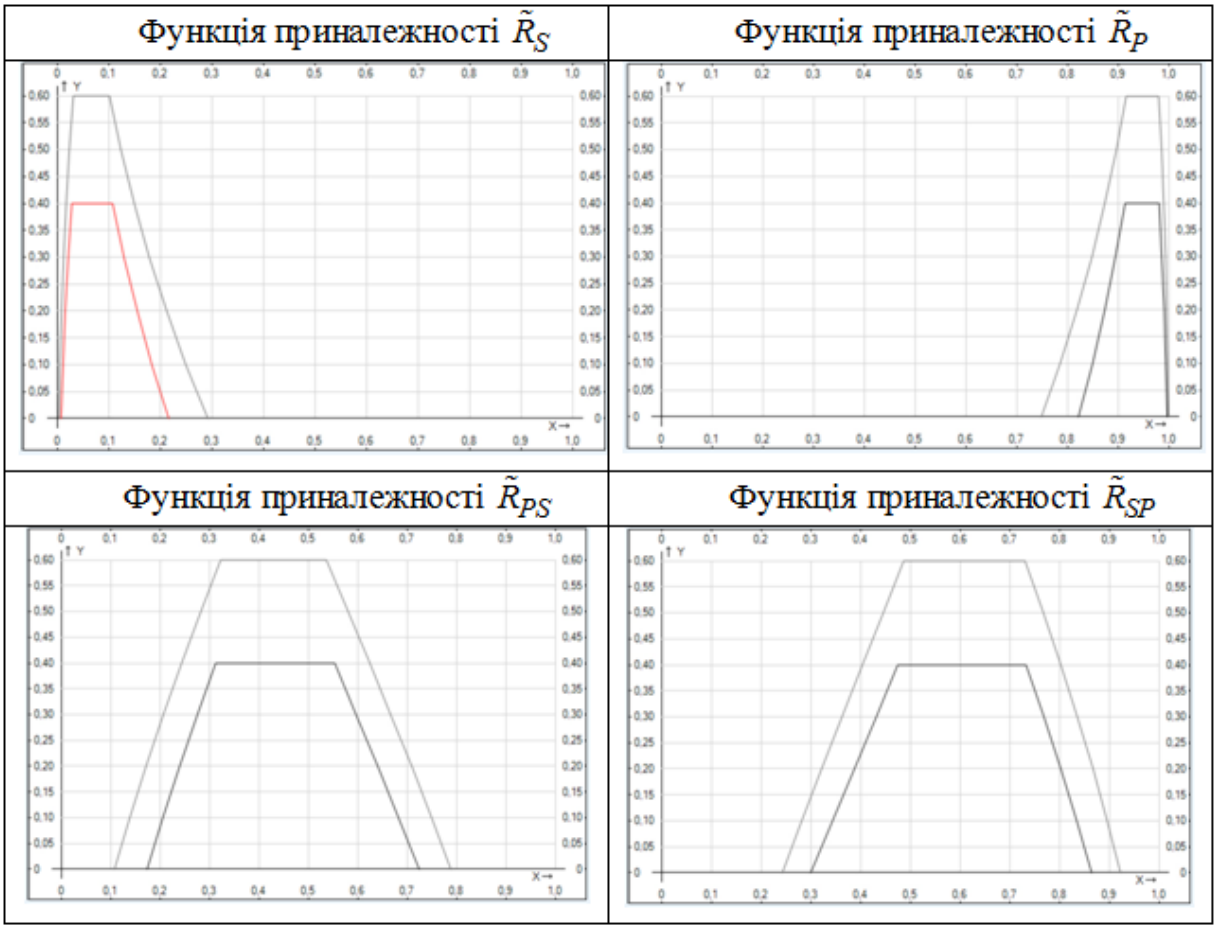


Рис. 4.29 Надійність нечітких систем з розмитими надійнісними параметрами компонентів

Далі наведені електронні таблиці з уніфікованим описом компонентів структур і розрахунковими формулами для обробки вихідних даних і оцінки показників надійності нечіткої системи з розмитими межами.

1. Вихідні дані компонентів $\tilde{R}_1, \tilde{R}_2, \tilde{R}_3, \tilde{R}_4$.

	B	C	D	E	F	G
3		R1	R2	R3	R4	
4	a1	0,2	0,3	0,6	0,2	
5	b12	0,4	0,5	0,7	0,3	
6	b22	0,4	0,7	0,7	0,4	
7	c1	0,6	0,9	0,8	0,5	
8	a2	0,2	0,3	0,5	0,1	
9	b21	0,4	0,5	0,7	0,3	
10	b22	0,4	0,7	0,7	0,4	
11	c2	0,6	0,9	0,9	0,6	min
12	alfa	0,7	0,6	0,5	0,4	**0,4**
13	betta	0,8	0,9	0,7	0,6	**0,6**

МИН(C12:F12)

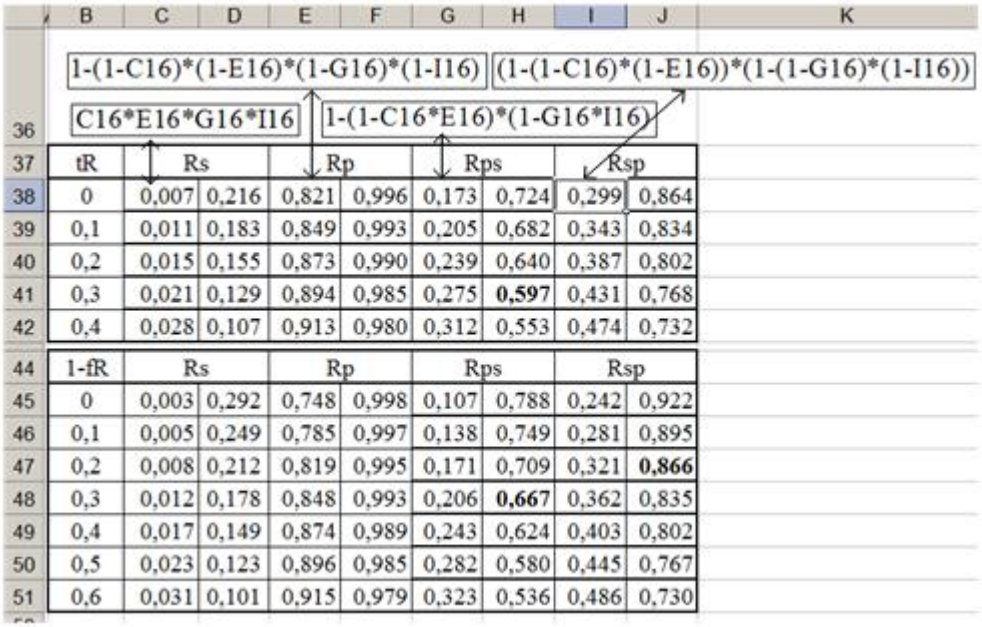
2. Обробка вихідних даних: побудова графіків і α, і β відтинання.

	B	C	D	E	F	G	H	I	J	K
14										
15	tR	R1		R2		R3		R4		
16	0	0,200	0,600	0,300	0,900	0,600	0,800	0,200	0,500	
17	0,1	0,229	0,571	0,333	0,867	0,620	0,780	0,225	0,475	
18	0,2	0,257	0,543	0,367	0,833	0,640	0,760	0,250	0,450	
19	0,3	0,286	0,514	0,400	0,800	0,660	0,740	0,275	0,425	
20	0,4	0,314	0,486	0,433	0,767	0,680	0,720	0,300	0,400	
21	**0,5**	**0,343**	**0,457**	**0,467**	**0,733**	**0,700**	**0,700**			
22	**0,6**	**0,371**	**0,429**	**0,500**	**0,700**					
23	**0,7**	**0,400**	**0,400**							
25	1-fR	R1		R2		R3		R4		
26	0	0,200	0,600	0,300	0,900	0,500	0,900	0,100	0,600	
27	0,1	0,225	0,575	0,322	0,878	0,529	0,871	0,133	0,567	
28	0,2	0,250	0,550	0,344	0,856	0,557	0,843	0,167	0,533	
29	0,3	0,275	0,525	0,367	0,833	0,586	0,814	0,200	0,500	
30	0,4	0,300	0,500	0,389	0,811	0,614	0,786	0,233	0,467	
31	0,5	0,325	0,475	0,411	0,789	0,643	0,757	0,267	0,433	
32	0,6	0,350	0,450	0,433	0,767	0,671	0,729	0,300	0,400	
33	**0,7**	**0,375**	**0,425**	**0,456**	**0,744**	**0,700**	**0,700**			
34	**0,8**	**0,400**	**0,400**	**0,478**	**0,722**					
35	**0,9**			**0,500**	**0,700**					

$F$7-B17*($F$7-$F$6)/$F$12

B17*($F$5-$F$4)/$F$12+$F$4

3. Розрахунок показників надійності (за формулами таблиці Н.1).



	B	C	D	E	F	G	H	I	J	K
	1-(1-C16)*(1-E16)*(1-G16)*(1-I16)							(1-(1-C16)*(1-E16))*(1-(1-G16)*(1-I16))		
36	C16*E16*G16*I16			1-(1-C16*E16)*(1-G16*I16)						
37	tR	Rs		Rp		Rps		Rsp		
38	0	0,007	0,216	0,821	0,996	0,173	0,724	0,299	0,864	
39	0,1	0,011	0,183	0,849	0,993	0,205	0,682	0,343	0,834	
40	0,2	0,015	0,155	0,873	0,990	0,239	0,640	0,387	0,802	
41	0,3	0,021	0,129	0,894	0,985	0,275	**0,597**	0,431	0,768	
42	0,4	0,028	0,107	0,913	0,980	0,312	0,553	0,474	0,732	
44	1-fR	Rs		Rp		Rps		Rsp		
45	0	0,003	0,292	0,748	0,998	0,107	0,788	0,242	0,922	
46	0,1	0,005	0,249	0,785	0,997	0,138	0,749	0,281	0,895	
47	0,2	0,008	0,212	0,819	0,995	0,171	0,709	0,321	**0,866**	
48	0,3	0,012	0,178	0,848	0,993	0,206	**0,667**	0,362	0,835	
49	0,4	0,017	0,149	0,874	0,989	0,243	0,624	0,403	0,802	
50	0,5	0,023	0,123	0,896	0,985	0,282	0,580	0,445	0,767	
51	0,6	0,031	0,101	0,915	0,979	0,323	0,536	0,486	0,730	

4.5 Інформаційно-часові діаграми для опису алгоритмів роботи децентралізованих інформаційних систем реального масштабу часу

Як показує досвід застосування мікропроцесорних засобів для автоматизації управління складними технологічними процесами (розділ 1), мікропроцесорні керуючі обчислювальні системи є ієрархічними багатофункціональними децентралізованими системами, що містять декілька функціонально-розподілених (можливо, і територіально-розподілених) мікроконтролерів (МКК) і (або) МІКРО-ЕОМ (МКЕОМ) з відповідним набором пристроїв зв'язку з виділеною частиною об'єкту управління і оперативно-диспетчерського устаткування [15, 107, 95, 48, 131].

У даній частині розглядається одне з достатньо важливих питань, пов'язаних з розробкою математичного забезпечення децентралізованих керуючих систем, а саме принципи опису загального алгоритму їх функціонування. У складі математичного забезпечення АСКТП це один з основних документів, в якому концентрується і узгоджується інформація

про алгоритми управління і контролю та їх інформаційне забезпечення. Цей документ значною мірою визначає «розуміння» роботи системи її розробниками, програмістами.

В той же час на етапі системотехнічного проектування спеціального математичного і інформаційного забезпечення АСКТП при описі загального алгоритму функціонування системи є певні труднощі. Необхідно представити роботу системи, що складається з декількох паралельно функціонуючих і взаємодіючих між собою підсистем, пов'язавши в реальному масштабі часу їх алгоритми контролю і управління. Крім того, слід врахувати різні режими роботи ТОК (послідовний, паралельний, з технологічними збоями і ін.), передбачити забезпечення відмовостійкої роботи системи. Виконаний аналіз літератури показав, що представлення загального алгоритму функціонування багатомашинних систем управління у вигляді таблиць, графів, текстового матеріалу або їх комбінацій не дає ясного уявлення про роботу систем зважаючи на громіздкість, багатослівність, недостатню структурованість.

Розглянемо принципи опису алгоритмів функціонування децентралізованих систем керування, розроблені для АСК маршрутами руху відчепів на сортувальній гірці, побудованій на базі чотирьох функціональних МКЕОМ [15, 98].

Початковими передумовами для розробки нового підходу явились «технології» опису систем, що проектуються, які розвивалися останнім часом, в першу чергу H1PO-технология [9, 149] і ідеї структурного програмування [46].

У основу опису загального алгоритму функціонування покладені наступні основні принципи.

1. Загальний алгоритм функціонування описується на основі наступного алфавіту (початкових даних):

- алгоритми контролю і керування, що мають шифр і коротке найменування, яке відображає, що робиться у відповідному алгоритмі;

- основні масиви і змінні, що використовуються і формуються

алгоритмами, мають ім'я (мітку);

- периферійні пристрої, основні датчики і виконавчі механізми, що використовуються в алгоритмах для вводу, виводу або зберігання інформації;

- інформаційні зв'язки алгоритмів з програмно-опитуваними сигналами, з каналами передачі даних, з периферійними пристроями.

2. Перераховані елементи і зв'язки графічно представляються у вигляді двох умовних схем:

- інформаційно-алгоритмічних графів - ІАГ (таблиць) - функціональних підсистем;

- функціональних часових діаграм (ФЧД).

ІАГ відображають алгоритмічний склад, інформаційне забезпечення і всі можливі зв'язки, що потенційно реалізовуються, всередині і поза функціональними підсистемами в процесі функціонування системи. ІАГ або замінююча його таблиця зв'язків дозволяє встановити для конкретного елементу з описаного складу всі його можливі зв'язки, які можуть мати місце в процесі функціонування системи. Це дозволяє проконтролювати по входу/виходу стиковку окремих алгоритмів, дає уявлення про місце конкретного алгоритму в комплексі вирішуваних системною задач (хоч би по його зв'язності). Окремі зв'язки (або позиції таблиці) можуть містити характеристику передаваної інформації (наприклад, об'єм повідомлення та ін.). ІАГ є збільшеною М-моделлю процесів керування (п. 3.1) і може використовуватися для структурного аналізу системи. На рис. 4.30 приведений фрагмент ІАГ загального алгоритму функціонування системи АСКМР.

На ФЧД відображена послідовність (динаміка) виконання різних алгоритмів, включаючи причину виникнення або ініціативу запуску, основні використовувані масиви, змінні, периферійні пристрої введення інформації, датчики (розташовуються над ФЧД) і основні масиви, змінні, що формуються і використовуються для введення даних, периферійні пристрої і виконавчі механізми (розташовуються під ФЧД). Для складання

ФЧД виконується структуризація процесів автоматизованого керування технологічним комплексом, тобто загальний алгоритм функціонування представляється у вигляді декількох основних взаємозв'язаних процесів (режимів) роботи системи, кожний з яких розбивається на послідовність технологічних етапів, що відбуваються у всіх підсистемах. Наприклад, в загальному алгоритмі функціонування АСКМР виділені наступні процеси.

Процес 1 - алгоритм функціонування системи при послідовному розпуску без технологічних збоїв (основний процес).

Процес 2 - алгоритм функціонування системи при паралельному розпуску без технологічних збоїв.

Процес 3 - алгоритм функціонування системи при послідовному (паралельному) розпуску з технологічними збоями.

Процес 4 - алгоритм функціонування системи в умовах відмов (збоїв) технічних засобів.

Процес 5 - алгоритм функціонування системи при виконанні маневрової роботи.

Кожний з виділених процесів розбитий на п'ять послідовних технологічних етапів:

- підготовка до розпуску - від прийому початкової інформації з АСК сортувальної станції (АСК СС) до початку розпуску (ЕТАП-П);

- початок розпуску - від натиснення кнопки «Пуск» до вступу першого відчепа составу, що розпускається, на ділянку контролю розчепа (ЕТАП-Н);

- контроль розчепа - рух відчепів по ділянці контролю розчепа (ЕТАП-Р);

- контроль і керування маршрутами відчепів, що скачуються, - рух відчепів по стрілочних колійних ділянках (СКД) до проходження останньої осі останнього відчепа через фіксований датчик останньої СКД або до натиснення кнопки «Кінець розпуску» (ЕТАП-К).

Умовні позначення на схемах наведені в таблиці 4.10.

У основу побудови ФЧД покладені наступні основні принципи (див. рис. 4.31):

Таблиця 4.10

Умовні позначення на ФЧД

	Сигнал, що обробляється програмно
	Ініціативний сигнал
	Міжмашинні канали
	Зв'язки з периферійними пристроями
	Зв'язки між алгоритмами

- ФЧД будуються для кожного етапу функціонування системи і включають всі процеси, що відбуваються на даному етапі у всіх відповідних підсистемах;

- на кожному з етапів виділяються ФЧД процесів, що реалізовуються в різних підсистемах;

- ФЧД відповідного етапу, що відносяться до конкретної підсистеми, включають основні елементи інформаційного забезпечення даної підсистеми на даному етапі, що використовуються або формуються, причому на верхньому полі цих ФЧД показані дані, що вводяться, і їх джерела, а на нижньому - дані, що виводяться, і їх приймачі;

- на ФЧД виділяються типи інформаційних управляючих зв'язків між алгоритмами і елементами інформаційного забезпечення або відповідними пристроями, а саме ініціативні сигнали, що поступають через ПЗО, через канали міжмашинного зв'язку, через канали зв'язку з периферійними пристроями, програмно-опитувані сигнали, інформаційні зв'язки алгоритмів з інформаційними масивами і змінними;

- алгоритми на ФЧД представляються прямокутниками довільної довжини, яка вибирається виходячи із зручності зображення діаграми; у кожному прямокутнику записуються шифр завдання або алгоритму і її скорочене найменування;

- всі алгоритми розташовуються на осі часу в черговості, яка визначається загальним алгоритмом функціонування системи. На осі часу для кожної підсистеми ставляться відносні мітки часу t_i, t_j, де i, j - десяткові числа, що відображають лише послідовність виконання алгоритмів, причому $t_i < t_j$, якщо $j > i$; якщо алгоритм E запускає алгоритм G, то закінченню блоку E відповідає початок блоку G в конкретний момент часу. У деяких алгоритмах виділяються окремі умови (W), що впливають на хід процесів контролю і керування в системі, в цьому випадку ФЧД розгалужуються і відображаються дві або більше можливих послідовностей виконання алгоритмів;

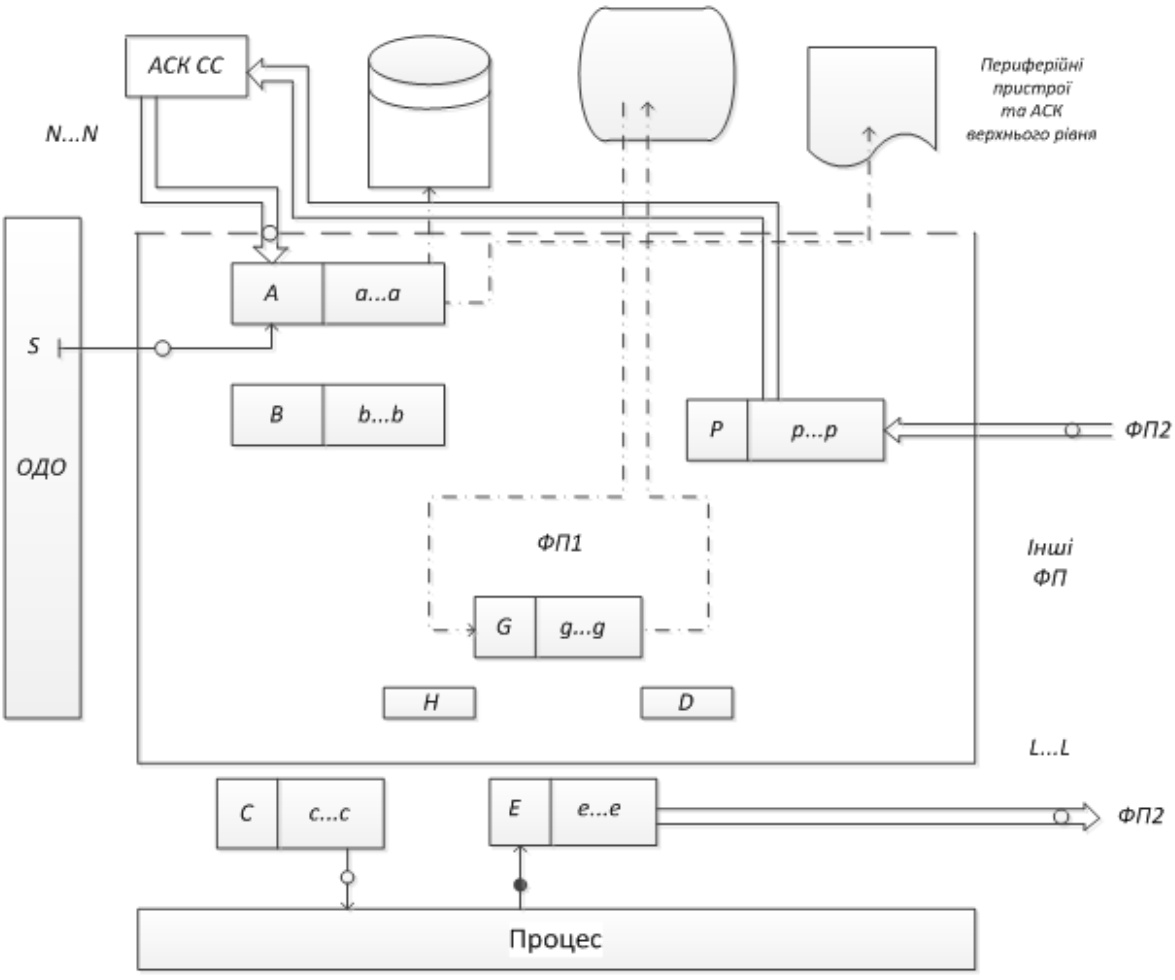


Рис. 4.30 Принципи побудови ФЧД

- для асинхронних процесів, що не мають жорсткої відносної прив'язки в часі, індекси міток часу позначаються символами, наприклад t_a, t_b.

Слід відмітити, що послідовності алгоритмів на ФЧД, що запускаються по конкретних ініціативних сигналах, в ході управління технологічним процесом багато разів повторюються, що визначається кількістю ініціативних сигналів відповідного типу, що поступають.

Приклад побудови ФЧД наданий на рисунку 4.31, а використовувані при цьому додаткові умовні позначення приведені в таблиці 4.10. Схема читається таким чином. Розглядається процес за номером 1, що реалізується в системі ФП1.

У момент часу t_i поступає ініціатива з АСК СС з передачею повідомлення N...N, по якій виконується алгоритм А (найменування «а...а» на ФЧД), що забезпечує: виконання операцій обробки даних; вивід на друк повідомлення Ф...Ф; введення інформації Q...Q з накопичувача на магнітних дисках (НМД); введення через ПЗО сигналу S; вивід на НМД даних R...R; аналіз умови W. При виконанні умови W починає роботу алгоритм В. Якщо умова W не виконується, запускається алгоритм С, що забезпечує видачу керуючого сигналу V...V;

$[t_{i+4}, t_{i+5}]$ - за ініціативою F виконується алгоритм Е, що забезпечує: формування масиву D; читання змісту масиву Н; передачу в іншу підсистему (ФП-2) інформації L...L, яка запускає алгоритм І;

$[t_{i+5}, t_{i+6}]$ - виконується алгоритм G, що забезпечує: введення з терміналу інформації Х...Х, обробку даних і видачу на термінал повідомлення Y...Y;

$[t_{i+7}, t_{i+8}]$ - за ініціативою від завдання К іншої підсистеми виконується алгоритм Р, що забезпечує підготовку даних і видачу до АСК СС повідомлення М...М.

Запропоновані принципи опису загального алгоритму функціонування, що використалися при розробці відповідної частини технічного проекту децентралізованої АСК МР, можуть знайти застосування при побудові загальних алгоритмів роботи інших децентралізованих інформаційно-керуючих систем АСК.

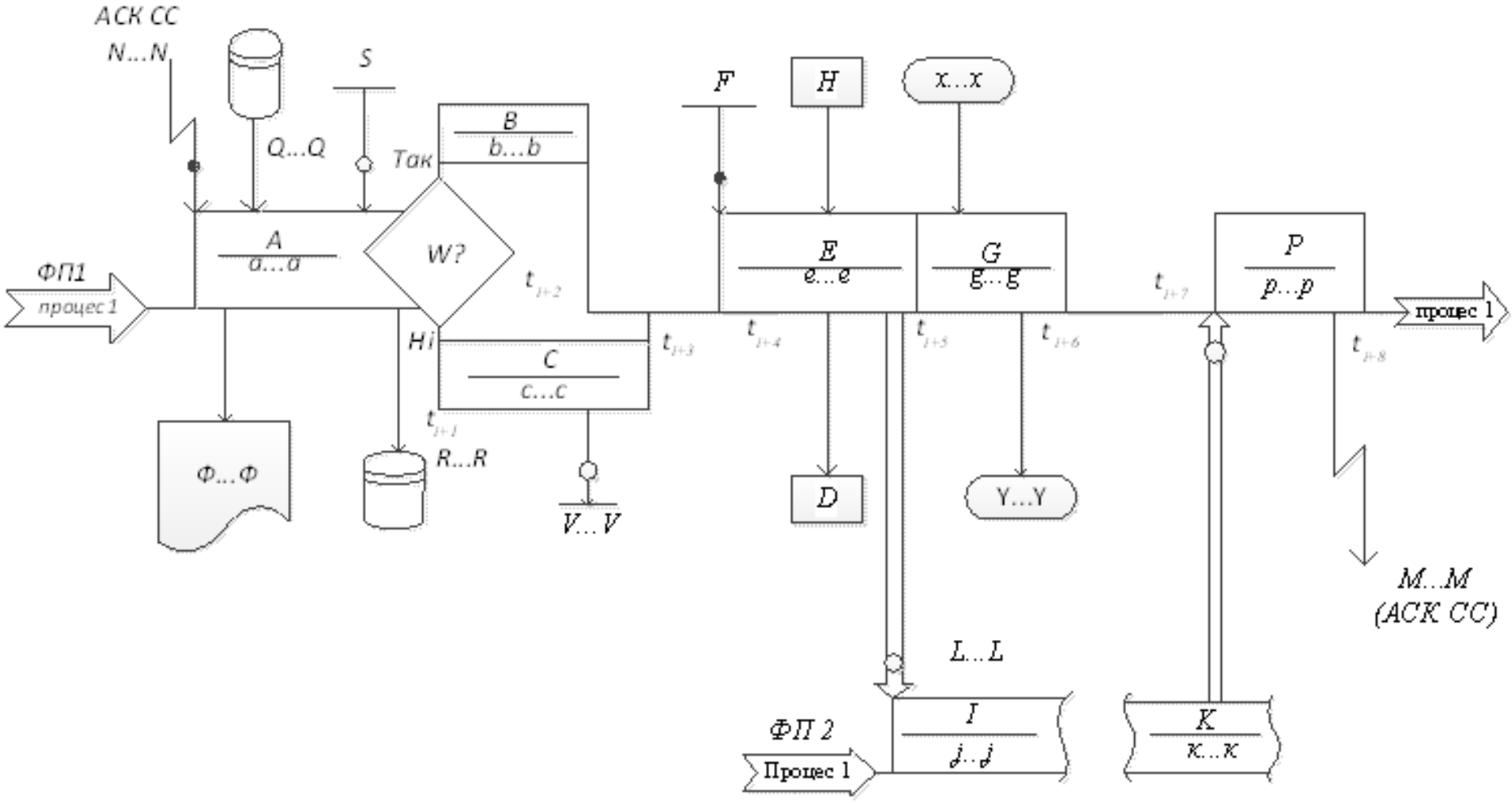


Рис. 4.31 Фрагмент IAГ загального алгоритму роботи системи АСКМР

4.6 Логіко-лінгвістична система керування сповільнювачем прицільної гальмівної позиції на сортувальній гірці

Управління швидкістю скочування відчепів на гірках сортувальних станцій є однією з найбільш складних задач. Вона вирішується в умовах неточності параметрів відчепів та умов їх скочування, які залежать від множини технологічних, технічних і погодних факторів. Незважаючи на застосування складних алгоритмів управління і сучасних ЕОМ і контролерів, проблема якісного управління залишається, як і раніше, актуальною [30]. В цих умовах, дії багатьох операторів, що керують гальмуванням відчепів на гальмівних позиціях, залишаються найбільш раціональними та ефективними. Розглянемо приклад управління відчепами на прицільній гальмівній позиції з використанням нечіткого контролера на основі нечіткого виводу (рис. 4.32).

Для управління сповільнювачем пропонується використовувати три вхідні лінгвістичні змінні: швидкість відчепа в сповільнювачі, його прискорення і точка прицілювання - відстань до найближчого відчепу, що стоїть на сортувальному шляху. Один вихідний параметр - ступінь гальмування 0, 1, 2, 3. (рис. 4.33).

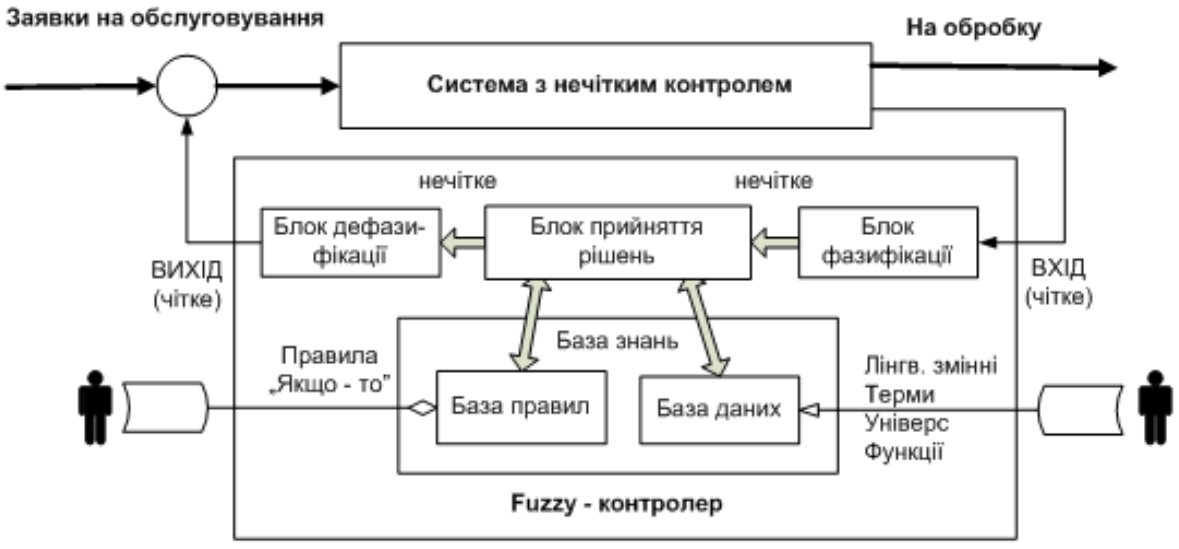


Рис. 4.32 Структура системи нечіткого виведення

В якості схеми нечіткого виводу будемо використовувати метод Мамдані, тому методом активізації буде MIN, для акумуляції висновків правил використовується метод MAX-диз'юнкції.

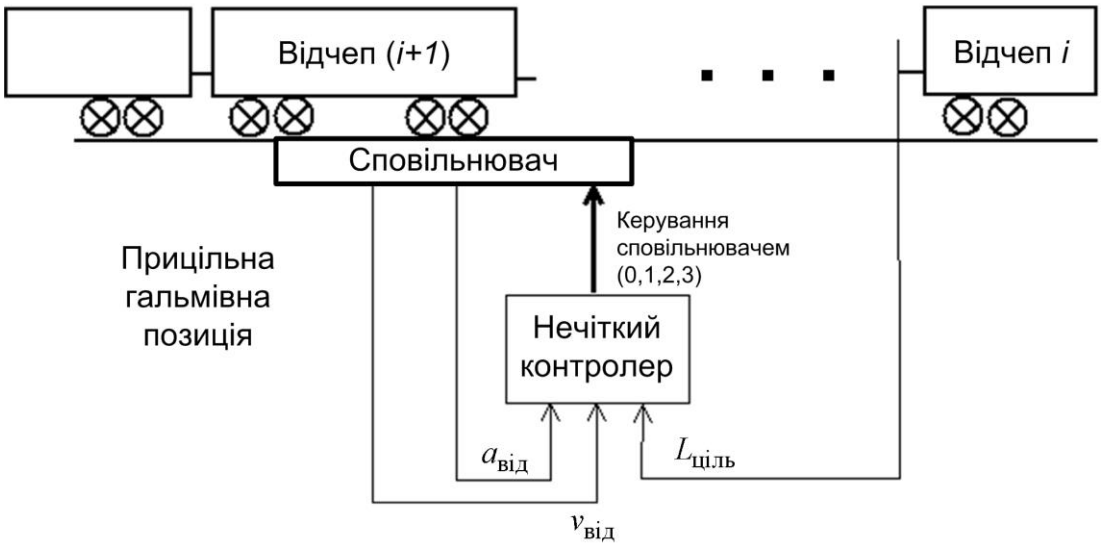


Рис. 4.33 Структурна схема керування сповільнювачем прицільної гальмівної позиції з використанням нечіткого контролера

В якості методу дефазифікації пропонується відомий метод центру тяжіння для одноелементних множин. Центр тяжіння (COGS, Centre of Gravity for Singletons) для одноточкових множин розраховується за формулою

$$z = \frac{\sum_{i=1}^{n} x_i \cdot \mu(x_i)}{\sum_{i=1}^{n} \mu(x_i)},$$

де n - число одноточкових (одноелементних) нечітких множин, кожне з яких характеризує єдине значення розглянутої вихідної лінгвістичної змінної [119].

Для кожного з трьох вхідних логіко-лінгвістичних змінних визначені універси (діапазони зміни параметрів), перелік відповідних термів та їх функції приналежності. У роботі використовується загальноприйнята система позначень термів, адаптована до української мови (таблиця 4.11) [110].

Таблиця 4.11

Загальноприйняті скорочення термів лінгвістичних змінних в системах нечіткого виводу

Символічне позначення	Англомовна нотація	Україномовна нотація	Україномовне символічне позначення
NB	Negative Big	Негативне велике	НВ
NM	Negative Middle	Негативне середнє	НС
NS	Negative Small	Негативне мале	НМ
ZN	Zero Negative	Негативне близьке до 0	НН
Z (ZO)	Zero	Нуль, близьке до 0	Н
ZP	Zero Positive	Позитивне близьке до 0	ПН
PS	Positive Small	Позитивне мале	ПМ
PM	Positive Middle	Позитивне середнє	ПС
PB	Positive Big	Позитивне велике	ПВ

Наприклад, для швидкості відчепа в сповільнювачі обрано терми [PS, PM, PB], для прискорення відчепа в сповільнювачі - [NS, Z, PS], для відстані до найближчого відчепу на сортувальному шляху - [PS, PM, PB]. Вихідна змінна, ступінь гальмування – чотири терми [Z, PS, PM, PB]. Для зазначених логіко-лінгвістичних змінних сформульовані 27 логіко-лінгвістичних продукційних правил завдання положення регулятора сповільнювача прицільної гальмівної позиції (таблиця 4.12).

Робота контролера нечіткого логічного виведення була промодельована в системі FLS [110] і досліджена при заданих універсах і функціях приналежності, при цьому були отримані задовільні результати.

Розроблену логіко-лінгвістичну систему можна вважати експериментальною. Для конкретних сортувальних гірок необхідно уточнювати

універси, кількість термів для подання вхідних змінних, вид їх функцій приналежності і відповідні правила логічного виводу, враховуючи досвід роботи висококваліфікованих операторів сортувальних гірок. Для цього

автором розроблена спеціальна імітаційно-аналітична методика налагодження нечіткого контролера.

Таблиця 4.12

Правила завдання положення регулятора

Номер правила	Вхідні лінгвістичні змінні			Вихідна лінгв. змінна
	Швидкість відчепа	Прискорення	Відстань цілі	Положення регулятора
1	PS	NS	PS	Z
2	PM	NS	PS	Z
3	PB	NS	PS	PS
4	PS	Z	PS	Z
5	PM	Z	PS	PS
6	PB	Z	PS	PM
7	PS	PS	PS	PS
8	PM	PS	PS	PM
9	PB	PS	PS	PB
10	PS	NS	PM	Z
11	PM	NS	PM	Z
12	PB	NS	PM	PS
13	PS	Z	PM	Z
14	PM	Z	PM	PS
15	PB	Z	PM	PM
16	PS	PS	PM	Z
17	PM	PS	PM	PS
18	PB	PS	PM	PM
19	PS	NS	PB	Z
20	PM	NS	PB	PS
21	PB	NS	PB	PS
22	PS	Z	PB	Z
23	PM	Z	PB	PS
24	PB	Z	PB	PM
25	PS	PS	PB	PS
26	PM	PS	PB	PM
27	PB	PS	PB	PB

4.7 Принципи нечіткої маршрутизації для розподілених систем і мереж

Система АСК ВП УЗ-Є інтегрує територіально-розподілені АСК нижніх рівнів керування (див. розділ 1), які взаємодіють між собою, використовуючи технологію комутації пакетів, коли дані у вигляді послідовності пакетів можуть слідувати різними шляхами до місця призначення. Проте ця архітектура не гарантує мінімальної затримки і невеликої ймовірності втрат пакетів. Передбачається, що наступне покоління АСК ВП УЗ-Є буде будуватися на наступному поколінні високошвидкісних мереж, що працюватимуть не з комутацією пакетів, а з комутацією каналів і передачею трафіку в режимі реального часу [177]. Це означає, що перед тим, як хост відправить перший пакет на інший вузол, встановлюється зв'язок (канал) між двома хостами і цей зв'язок не змінюється згодом. Канал по дорозі від відправника до місця призначення проходить через ряд проміжних маршрутизаторів і ліній зв'язку.

Проблема маршрутизації полягає у виборі шляху з достатніми ресурсами для задоволення вимог за якістю сервісів (QoS, Quality of Service - якість обслуговування) у кожному елементарному з'єднанні з використанням інформації про стан мережі. Стан шляху зазвичай визначається станами маршрутизаторів і ліній зв'язку, а саме доступною (вільною) смугою пропускання, об'ємом буферної пам'яті і чергами і затримками передачі пакетів. Проблема маршрутизації може розглядатися як для індивідуального трафіку (unicast), так і для мультитрафіка (multicast) залежно від кількості вузлів призначення. Огляд алгоритмів маршрутизації можна знайти в [177].

Оскільки багато параметрів в алгоритмах маршрутизації є невизначеними або складні в аналітичній обробці, розглянемо нечіткий алгоритм маршрутизації [216].

Розглянемо мережу, представлену на рис. 4.34. Для простоти передбачається, що всі зв'язки мають однакову пропускну спроможність (100 Мбіт) і довжину (2 км.).

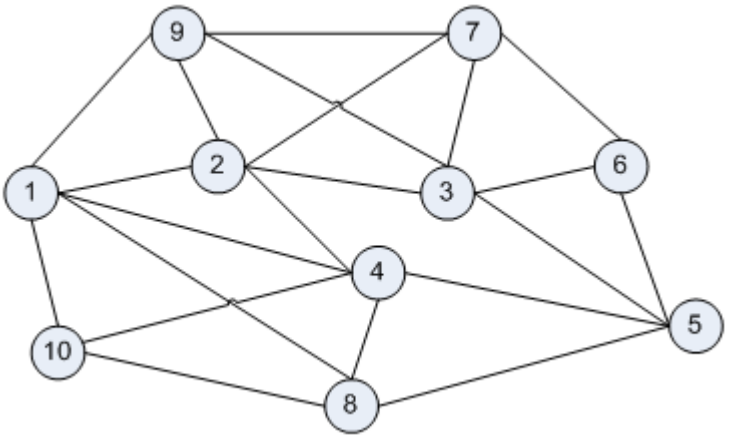


Рис. 4.34 Комунікаційна мережа

Ці припущення логічні тому що, як правило, затримки розповсюдження по каналах зв'язку значно менше затримок в чергах в комутаційних вузлах. Хай кожна вершина має буфер вхідних пакетів максимальної місткості (60 пакетів). Вузли 1, 5, 6, 7, 8, 9 і 10 по периметру мережі діють як вузли генератори, одержувачі і перемикачі трафіку. Вузли 2, 3 і 4 - чисті вузли перемикачі.

Відповідно до вимог QoS шлях повинен бути визначений до посилки повідомлень з генеруючого вузла і вибраний маршрут не повинен надалі змінюватися. Проблема полягає у визначенні оптимальної політики маршрутизації для кожного потоку трафіку в конкретному вузлі генерації на основі стану системи для мінімізації затримок пакетів, втрат пакетів і розривів з'єднань в генеруючих вузлах.

Для будь-якої пари вузлів "джерело-приймач" стан кожного прийнятного шляху описується кількістю пакетів в черзі в кожному буфері по цьому шляху. Проте, цей стан змінюється, оскільки в кожному вузлі прибувають нові і йдуть обслужені пакети.

Вирішальні правила вибору маршруту застосовуються за часом тоді, коли генерується нова сесія або потік трафіку [216].

Щоб вирішити проблему, пропонується використовувати розмиту логіку для визначення рейтингу шляху, заснованого на нижчеописаному критерії для всіх існуючих шляхів між парою "джерело-приймач". Шлях з

найвищим рейтингом вибирається для передачі потоку трафіку. З'єднання розривається тільки тоді, коли всі буфери у вибраному шляху повністю переповнені. Інакше всі пакети сесії прямують по даному шляху. Пакети, що прибувають в переповнені буфери, - втрачаються.

Розглянемо шлях з s зв'язками і відповідними буферами $i = 0,1,2,...,s$. Завантаження ρ_i кожного буфера i на шляху визначається як зайнята частина буфера.

Якщо n_i - розмір черги в буфері i і B_i - місткість цього буфера, то

$$\rho_i = \frac{n_i}{B_i}. \qquad (4.55)$$

Далі, ми візьмемо суму цих завантажень, які можна зміряти, і згенеруємо вагу w_i для кожного буфера i

$$w_i = \frac{\rho_i}{\sum_{j=1}^{s} \rho_j} = \frac{\rho_i}{\sum_{j=1}^{s} \frac{n_j}{B_j}}. \qquad (4.56)$$

Тепер визначимо завантаження мережі R як зважену суму завантажених частин буферів, що належать шлях

$$R = \sum_{i=1}^{s} w_i n_i . \qquad (4.57)$$

Вираз (4.623) можна декілька спростити, якщо прийняти, що всі буфери мають однакову місткість B. Тоді (4.60 визначатиметься як

$$\rho_i = \frac{n_i}{B}. \qquad (4.58)$$

Тоді вага маршруту обчислюється по новому виразу

$$w_i = \frac{\rho_i}{\sum_{j=1}^{s} \rho_j} = \frac{\rho_i}{\sum_{j=1}^{s} \frac{n_j}{B}} = \frac{\rho_i}{\frac{n_1}{B} + \frac{n_2}{B} + ... + \frac{n_s}{B}} = \frac{B\rho_i}{\sum_{j=1}^{s} n_j}. \qquad (4.59)$$

Підставивши (4.63) в (4.64), отримаємо

$$w_i = \frac{n_i}{\sum_{j=1}^{s} n_j}. \qquad (4.60)$$

Очевидно, що вага маршруту - це середнє зважена довжина черги пакетів по даному шляху. Його завантаження R рахуємо по тій же формулі (4.62).

Для кожного шляху ми вибираємо кількість роутерів s у маршруті і його завантаження R як нечіткі вхідні змінні і визначаємо рейтинг шляху r як вихідну змінну алгоритму.

Всі змінні представлені чотирма термами ZO, PS, PM и PB [110]. Рейтинг шляху визначається як функція від s і R.

База нечітких правил для системи маршрутизації показана в таблиці 4.13.

Універс (universe) для всіх введених нечітких змінних - множина [0, 6]. Функції приналежності для s і r показані на рис. М.1. Відмітимо, що R є глобальною характеристикою зайнятості буферної пам'яті, а загальна затримка є функцією від розміру черги n_i; таким чином, вона збільшується як послідовність $1+2...+n_i$, сума елементів цього ряду для заданого i визначається $n_i(n_i+1)/2$. Звідси отримуємо послідовність сум 1, 3, 6, …

З цього виходить, чому вибрана функція приналежності для R, показана на рис. 4.36 [216]. В цілому величина R нормована до вибраного діапазону [0, 6].

Нечіткий алгоритм маршрутизації складається з наступних кроків:

1. У кожен період прийняття рішень для кожної пари вузлів "джерело-приймач" визначаються всі можливі шляхи між ними. Також записується інформація про стан черг у вузлах по кожному маршруту (n_i).

2. Для кожного шляху обчислюються значення для s і R з використанням виразів (4.60), (4.61), (4.62) або (4.82), (4.63), (4.65). Потім визначається рейтинг r через фазіфікацію, логічне виведення і

дефазіфікацію.

3. Отриманий шлях з найвищим рейтингом ми вибираємо для передачі потоку трафіку.

Порівняння запропонованого алгоритму нечіткої маршрутизації з алгоритмами фіксованої маршрутизації і мінімального шляху показало, що він не поступається цим алгоритмам за критеріями мінімальної затримки і мінімальної ймовірності втрати пакетів [216].

Таблиця 4.13

База нечітких правил для системи маршрутизації [216]

Номер правила	Вхідні лінгвістичні змінні		Вихідна лінгв. змінна
	s	R	r
1	ZO	ZO	PB
2	PS	ZO	PM
3	PM	ZO	PS
4	PB	ZO	ZO
5	ZO	PS	PM
6	PS	PS	PS
7	PM	PS	ZO
8	PB	PS	ZO
9	ZO	PM	PS
10	PS	PM	ZO
11	PM	PM	ZO
12	PB	PM	ZO
13	ZO	PB	ZO
14	PS	PB	ZO
15	PM	PB	ZO
16	PB	PB	ZO

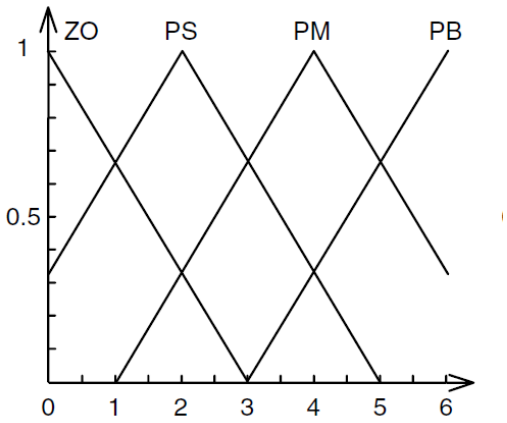


Рис. 4.35 Функції приналежності для s і r

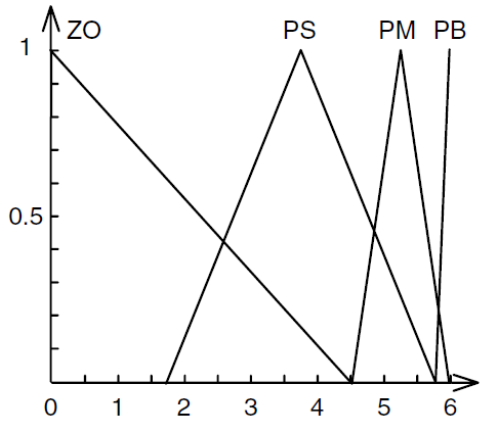


Рис. 4.36 Функції приналежності для R

Запропоновані принципи маршрутизації можна використовувати при виборі найкращих маршрутів перевезення вантажів автомобільним, залізничним і морським транспортом і в інших мережах масового обслуговування з чергами (з відповідним коректуванням вхідних параметрів, функцій приналежності і бази нечітких правил).

4.8 Онтологічні моделі в задачах автоматизації сортувальних станцій

Розробка систем автоматизації сортувальних станцій і гірок, як показали дослідження (розділ 1), пов'язана з необхідністю врахування великої кількості різноманітних параметрів, що зветься "прокляттям

розмірності". Перелічені особливості призвели до створення великої кількості несумісних підсистем автоматизації сортувальних станцій, які відрізняються назвами підсистем, функцій, складом вирішуваних завдань, технічними структурами, тощо.

Для подолання зазначених проблем пропонується створити онтологічні бази знань сортувальних станцій як об'єктів автоматизації та інших елементів архітектури систем.

Термін "онтологія" вперше з'явився в роботі Томаса Грубера, в якій розглядалися різні аспекти взаємодії інтелектуальних систем між собою і з людиною [186]. Інтелектуальні системи - це програми, які моделюють інтелектуальну діяльність людини. Знання, які закладає в програму її автор (тобто алгоритм цієї програми), завжди статичні, вони не змінюються. Інтелектуальна система в цьому сенсі більш універсальна - в ній знання про те, що треба робити в процесі виконання програми, не закладені в програму раз і назавжди, а можуть змінюватися. Якщо так, то ці знання необхідно передавати програмі як дані, тобто виникає необхідність їх опису.

Знання в комп'ютерних програмах, можна розділити на два типи: процедурні знання, тобто знання про те, що треба зробити в кожній конкретній ситуації і знання про проблемну область задачі або декларативні знання

Ідея Томаса Грубера полягала в тому, щоб дозволити інтелектуальним системам обмінюватися між собою закладеними в них знаннями щодо предметної області. Якщо всередині інтелектуальної системи знання про предметну область можуть бути закодовані як завгодно, то для обміну цими знаннями з іншою інтелектуальною системою необхідно надати опис цих знань. Цей опис має бути достатньою мірою формальним, щоб бути зрозумілим іншій системі, а також повинна бути відома мова цього опису. Крім того, опис має бути зрозумілим також і людині. Для цього Грубер запропонував описувати знання двома

способами: у канонічній формі, яка представляє собою опис знань на мові логіки предикатів; у формі онтології, яка являє собою множину класів, пов'язаних між собою відношенням узагальнення (це зворотне відношення для відносини спадкування).

Таким чином, онтологія за Грубером являє собою опис декларативних знань у вигляді класів та їх ієрархії. До цього опису, призначеному для читання людиною, додано опис в канонічній формі, що призначений для читання машинами. Кожна інтелектуальна система може надавати кілька таких описів, що відповідають різним областям декларативних знань і, таким чином, виступає як сховище бібліотеки онтологій. Грубер уявляв, що інтелектуальні системи будуть виступати як бібліотеки онтологій і зможуть вільно обмінюватися онтологіями між собою. При цьому бібліотеці онтологій вже не обов'язково бути інтелектуальною системою, досить просто надавати сервіс з передачі онтологій на вимогу.

Складання опису декларативних знань зазвичай вимагає великої роботи і певних навичок. Для позначення цієї роботи, а також її результату, Грубер ввів спеціальний термін «концептуалізація». Опис він називав «специфікацією». Таким чином, онтологія по Груберу визначається як специфікація концептуалізації [186].

Математично онтологія може бути представлена у вигляді кортежу:

$$O =< C, Pr, V, I, R, F >,$$

де

C - множина класів концептів предметної області; Pr - множина властивостей класів; V - множина значень властивостей; I - множина екземплярів класів; R - множина відносин між класами; F - множина аксіом, заданих на класах концептів.

Концептуалізація сортувальних станцій як об'єктів автоматизації являтиме собою основу першої базової онтології в бібліотеці онтологій для

проектування і удосконалення автоматизованих систем керування на сортувальних станціях (див. розділ 2).

Сортувальні станції, як підприємства по переробці вагонопотоків на мережі залізниць, можуть бути представлені у вигляді трьох макрорівнів.

Єдиний технологічний процес переробки потягів на сортувальних станціях можна представити у вигляді множини основних технологічних процесів.

Кожний технологічний процес P_{j} складається з множини технологічних операцій $P_l = \left\{ p_{il} \mid i = \overline{1, \mid P_l \mid}; j \in \{\text{пп,сг,сп,пв}\} \right\}$, що виконуються з використанням наземного устаткування і пристроїв низової автоматики.

В розділі 2 введено поняття технологічної ділянки, на якій виконується обмежений набір технологічних операцій з використанням відповідного комплекту наземного устаткування і пристроїв низової автоматики. Кожна технологічна ділянка має певне місце розташування на сортувальній станції, де здійснюється взаємодія з об'єктами керування (потягами, відчепами, вагонами). Для сортувальної станції N можна виділити кінцеву множину таких технологічних ділянок

$$Y_{\text{N}} = \left\{ y_1, y_2, ... y_{|Y_{\text{N}}|} \right\}. \tag{4.61}$$

Очевидно, що структуру будь-якого технологічного процесу можна описати як

$$\forall (P_l \in P_{\text{cc}}) \;\; P_l \subset Y_{\text{N}} \times Y_{\text{N}}. \tag{4.62}$$

Сортувальні станції зазвичай мають свою індивідуальну оснащеність наземним устаткуванням, приладами, низовою автоматикою і виконавчими механізмами. Всі види устаткування позначимо як деяка кінцева множина $O = \{o_1, o_2, ..., o_{|O|}\}$. Це устаткування розподілене по технологічних ділянках Y

.

Комплектування кожної ділянки можна описати виразом

$$\forall(y_j \in Y_{\mathrm{N}}) \quad O(y_j) \subset O \times O\,. \tag{4.63}$$

Кожний вид устаткування і пристроїв низової автоматики має свої засоби взаємодії з інформаційною інфраструктурою і з оперативним персоналом. Ми розглядатимемо тільки перший вид засобів, який визначає принципи формування інфраструктури, функції, що реалізовуються нею, і ефективність роботи системи в цілому.

Кожному виду устаткування відповідає кінцева множина сигналів і повідомлень, розподілених в часі

$$S_o = S^{in} \times S^{out} \times Msg^{in} \times Msg^{out} \times T \times F^{tr}\,. \tag{4.64}$$

Ініціативні вхідні сигнали, пов'язані з множиною дискретних подій (e, t_1, t_2) [82], які визначають динаміку функціонування підсистем інфраструктури і всієї системи в цілому.

Елементи множин S^{in}, S^{out} відрізняються електричними параметрами і схемами підключення.

Для будь-якої ділянки y_j комплект встановленого і використовуємого на нім устаткування $O(y_j)$ визначається набором вхідних-вихідних сигналів на даній ділянці $S(y_j) \subset S_o$.

Розглянемо основні етапи побудови онтологій автоматизованих сортувальних станцій.

Макрорівні станції можна представити у вигляді трьох концептів (класів або множин) на мові OWL в форматі OWL/XML. Для цього будемо використовувати відкритий редактор онтологій Protégé [162]. Формуємо ієрархію класів (рисунок 4.37).

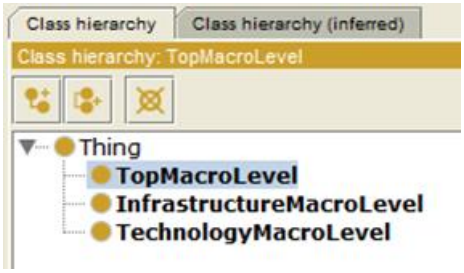


Рис. 4.37 Ієрархія класів

Завдання, які вирішуються на кожному иакрорівні, можна описати у вигдяді анотацій до кожного каласу.

Деталізуємо ієрархію класів сортувальної станції (див. рис. 4.38).

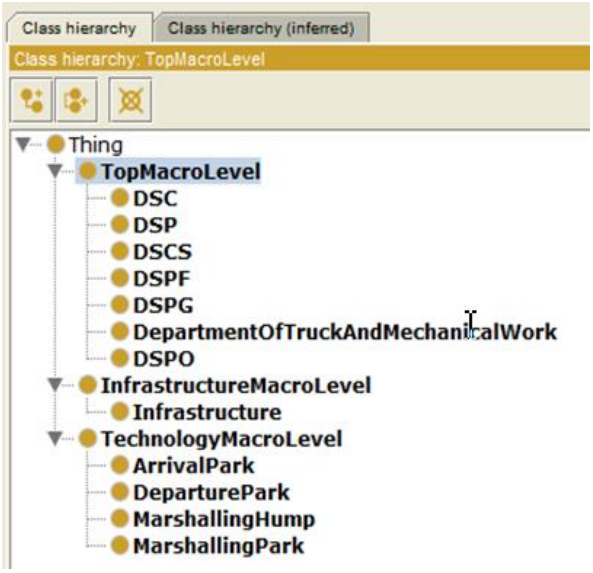


Рис. 4.38 Ієрархія класів сортувальної станції

Для представлення взаємозв’язків у базі знань використано властивості об’єкта - Object Properties. Властивості в OWL представляють відносини. Існує два основних типи властивостей: властивості об'єктів і властивості типу даних (DataType). Властивостями об'єкта є відносини між двома індивідами. Властивості об'єкта пов'язують індивідів. OWL також має третій тип властивості - властивості анотації. Властивості анотації можуть використовуватися для додавання інформації (метаданих) для класів, окремих індивідів і властивостей об'єктів (типів) даних.

Створення екземплярів Object Property аналогічно створенню ієрархії класів. Варіант ієрархії TopObjectProperty представлено на рисунку 4.39.

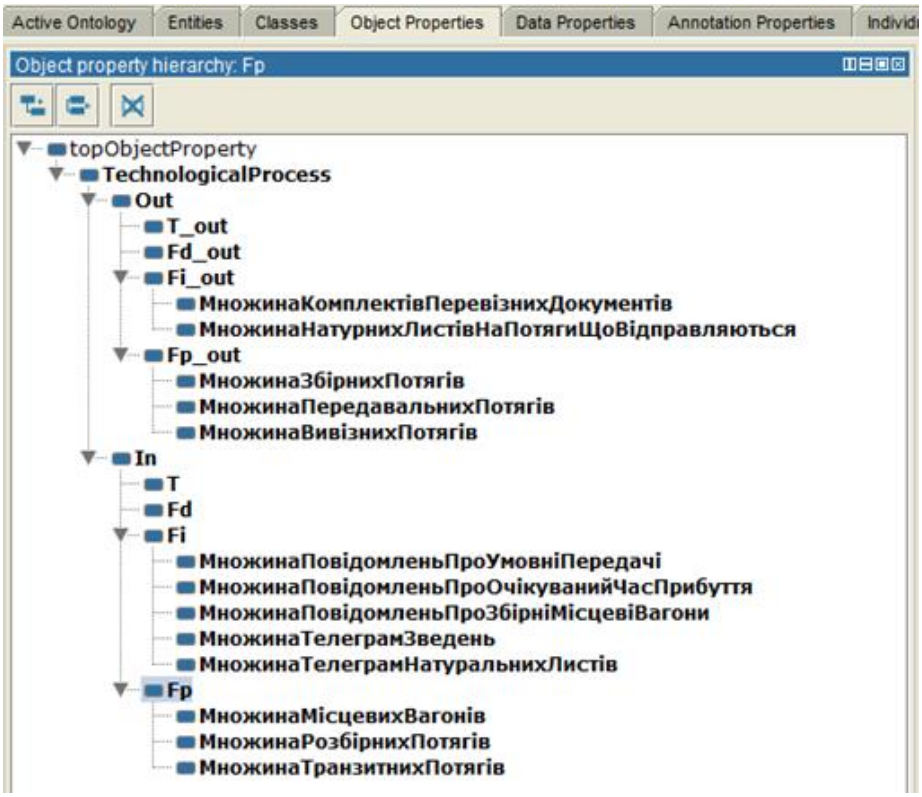


Рис. 4.39 Ієрархія TopObjectProperty

Розроблена концептуалізація автоматизованих сортувальних станцій до рівня сигналів дозволяє при розробці технічної структури системи сформувати уніфіковане проектне рішення на основі онтологічних моделей технічних засобів автоматизації, наприклад промислових комп'ютерів. На рисунку 4.40 приведена розроблена база знань системи індустріальних контролерів АРАХ фірми Advantech [158]

Онтології сортувальних станцій як об'єктів автоматизації і промислових контролерів АРАХ-5000 увійшли до бібліотеки онтологій системи автоматизованого проектування і удосконалення автоматизованих систем керування на сортувальних станціях.

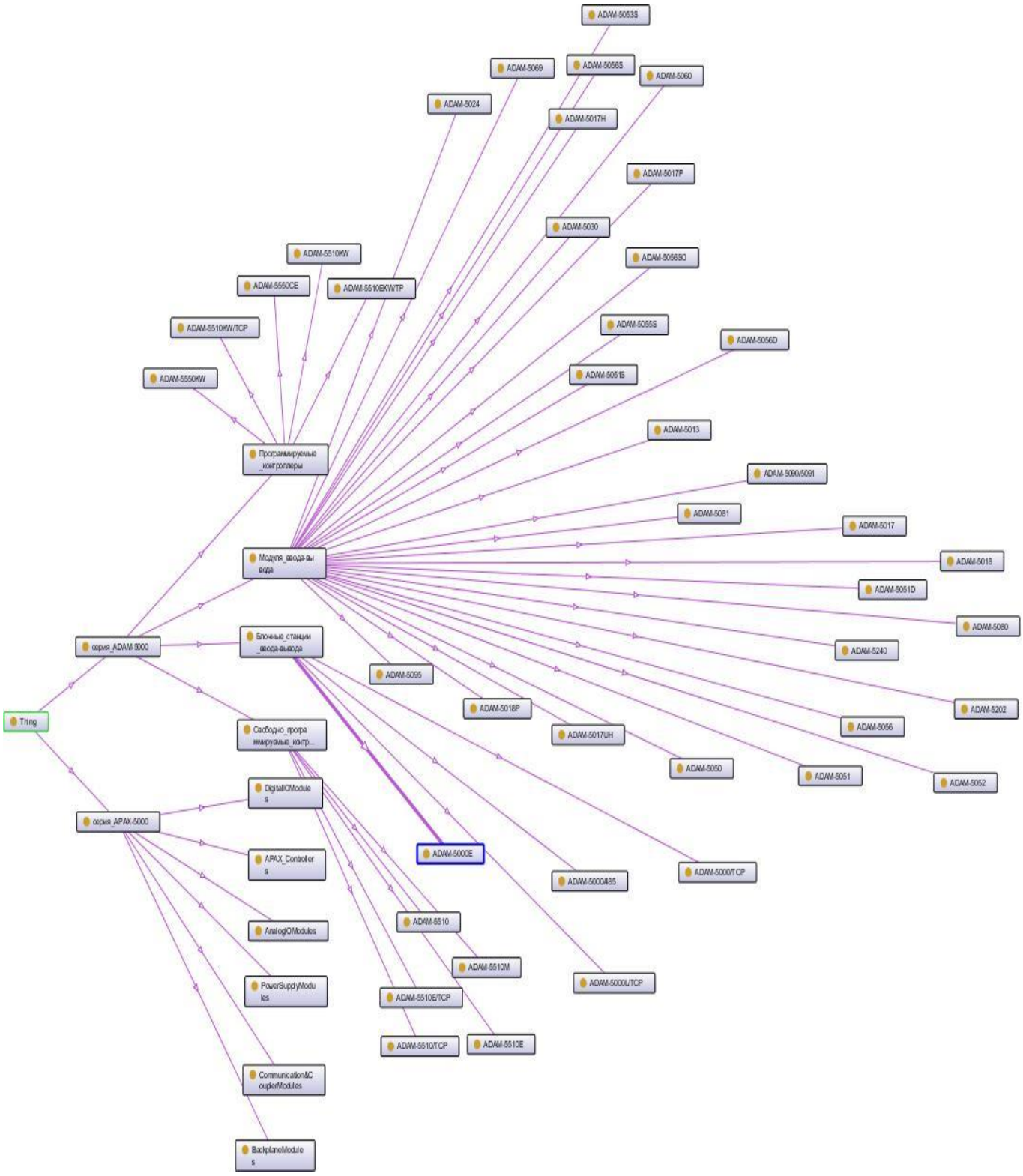


Рис. 4.40 Онтологія контролерів APAX

4.9 Інтелектуальні методи ідентифікації зібраних маршрутів в парках сортувальної станції

4.9.1. Основні положення

Однією з найважливіших задач інформаційно-керуючої системи парку відправлення сортувальної станції є контроль переміщень рухомих одиниць (поїздів, локомотивів і ін.) у межах парку. Така задача вирішується спеціальною підсистемою контролю переміщень. Для здійснення такого контролю в керуючу ЕОМ через пристрої сполучення з об'єктом (УСО) вводяться сигнали з напольного устаткування (рейкових кіл, точкових шляхових датчиків, контрольних реле стрілочних переводів і світлофорів і ін.). У процесі контролю переміщення об'єкта виникає задача ідентифікації зібраного маршруту. Як правило, маршрут містить у собі ряд ізольованих ділянок (рис. 4.41).

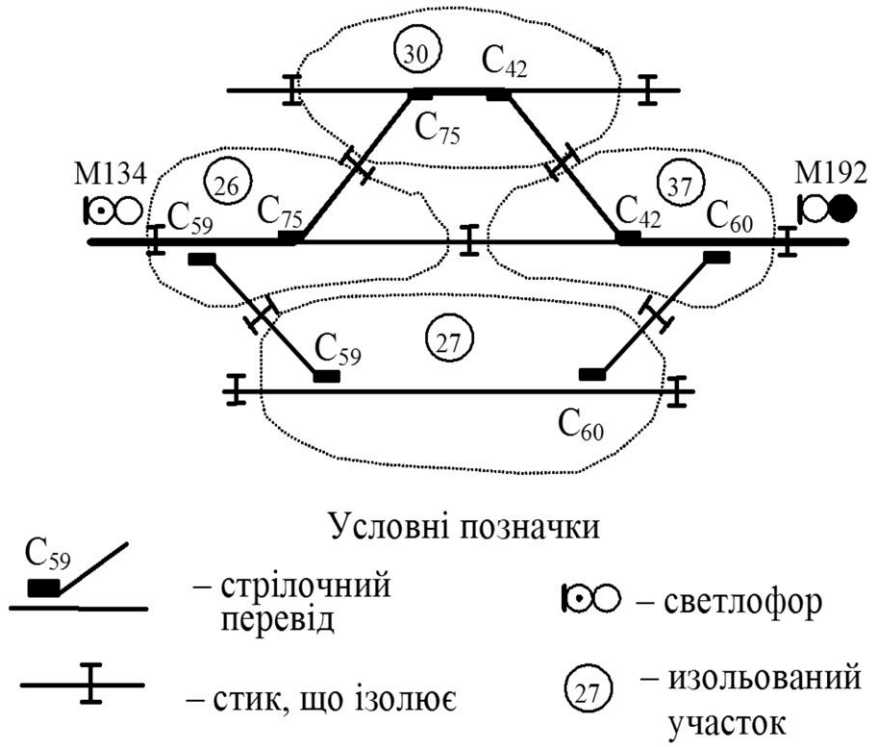


Рис. 4.41. Фрагмент сортувального парку

При цьому на ізольованій ділянці може бути кілька стрілочних перекладів. Початок і кінець маршруту визначають сигнали світлофора, що дозволяють і забороняють.

Традиційний алгоритм визначення зібраного маршруту в підсистемах контролю переміщень наступний. Починаючи з першої ділянки маршруту (першу ділянку визначають по приходу сигналу, що дозволяє, відповідного світлофора) по положенню стрілок на цій ділянці визначають номер ділянки маршруту, що примикає, (на рис. 4.42 це ділянка №2). Вважають цю ділянку включеною у маршрут. По положенню стрілок на цій ділянці визначають наступну ділянку, що примикає, і так далі поки на черговій ділянці, що примикає, не буде зафіксований сигнал світлофора, що забороняє. Якщо при цьому відбудеться перекручування хоча б одного сигналу про положення стрілки, то зібраний маршрут буде визначений невірно.

Будемо вважати імовірність перекручування сигналу про положення кожної стрілки однакової і рівний P_C. Тоді імовірність безпомилкового визначення маршруту

$$P_M = (1\text{-}P_C)^n , \qquad (4.65)$$

де n – кількість стрілок у маршруті.

Досвід експлуатації подібних систем показав, що величина P_M може бути неприйнятно мала: 95…97%. Таким чином, виникає задача підвищення вірогідності ідентифікації зібраних маршрутів. При цьому можливі два шляхи підвищення вірогідності:

1) Зменшення імовірності перекручування сигналу про положення стрілки. Цей шлях припускає додаткові апаратні витрати (використання більш надійних джерел сигналів і каналів зв'язку, або дублювання сигналів).

2) Використання спеціальних алгоритмів підвищення надійності, таких, щоб перекручування одного із сигналів не приводило до

гарантованого перекручування номера зібраного маршруту. Цей шлях припускає використання існуючих сигналів від наземного устаткування.

Тому що другий шлях виключає додаткові апаратні витрати розглянемо можливості алгоритмічного підвищення вірогідності ідентифікації зібраних маршрутів.

4.9.2. Використання додаткових сигналів від наземного устаткування

В якості додаткових можуть бути використані сигнали від так званих "замикаючих реле" (ЗР). Таке реле мається для кожної ізольованої ділянки. Звичайно це реле знаходиться під струмом. Але коли ізольована ділянка попадає в зібраний маршрут, то відповідне ЗР знеструмлюється. Однак сигнали ЗР не дублюють сигнали про положення стрілок, а лише доповнюють їх. Крім того, можуть бути зібрані кілька маршрутів, що використовують суміжні ізольовані ділянки. У такому випадку сигнал про знеструмлення ЗР не говорить прямо до якого маршруту відноситься дана ділянка. Очевидно, що лише правильне використання сигналів ЗР і сигналів положення стрілок може дати найбільше достовірну інформацію про номер зібраного маршруту.

4.9.3 Алгоритм підвищення ймовірності ідентифікації зібраних маршрутів

Кожен маршрут M (M=1…M_{max}) містить у собі деяку безліч стрілок $C_м \in \boldsymbol{З}$ ($\boldsymbol{З}$ – загальна безліч стрілок) і ізольованих ділянок $\theta_м \in \boldsymbol{\Theta}$ ($\boldsymbol{\Theta}$ – загальна безліч ізольованих ділянок). Для кожного маршруту повинне бути визначене сумарне число подій $K_м$, що відповідають ситуації, при якій даний маршрут вважається зібраним:

$$K_м = \sum_{\substack{i \\ c_i \in c_м}} c_i + \sum_{\substack{j \\ \theta_j \in \theta_м}} \theta_j , \qquad (4.66)$$

тут c_i – стрілки, що належать безлічі $\boldsymbol{C_м}$ (фактично $\sum_{\substack{i \\ c_i \in c_м}} c_i$ – це кількість стрілок у маршруті M); θ_j – ізольовані ділянки, що належать безлічі $\theta_м$ (фактично $\sum_{\substack{j \\ \theta_j \in \theta_м}} \theta_j$ – це кількість ізольованих ділянок у маршруті M).

Масив $K_м$ створюється заздалегідь (статичний метод формування масиву), або формується в момент надходження в систему сигналу світлофора, що дозволяє, (динамічний метод формування масиву).

Як приклад наведена таблиця 4.14 заданого числа подій (статичний метод формування масиву $K_м$) для фрагмента ділянки (рис. 4.42) парку відправлення сортувальної станції Нижньодніпровськ-Вузол.

Таблиця 4.14

Формування масиву заданої кількості подій

Мар-шрут	Ділянки								**Задана кіль-кість подій**	
	26		**27**		**30**		**37**			
	$З_{59}$	$З_{75}$	$З_{59}$	$З_{60}$	$З_7$	$З_{42}$	$З_{42}$	$З_{60}$	Стр. ЗР ∑	Σ
М1	0	×	0	0			×	0	2 3 5	5
М2	1	1					1	1	4 2 6	6
М3	1	0			0	0	0	1	4 3 7	7

Примітка: 0 – стрілка притиснута; 1 – стрілка віджата.
Заштриховано комірки для стрілок, що уже враховані на інших ділянках

Передбачається, що сигнали від стрілок і ЗР, що надійшли в систему, можуть бути перекручені. Тому при надходженні в систему сигналу світлофора, що дозволяє, аналізуються всі можливі маршрути, що починаються від ділянки з цим світлофором. При цьому для кожного

маршруту підраховують оцінку $\hat{K}_м$ подій, що відповідають цьому маршруту, за результатами аналізу сигналів, що прийшли в систему:

$$\hat{K}_м = \sum_{\substack{i \\ c_i \in c_м}} \hat{c}_i + \sum_{\substack{j \\ \theta_j \in \theta_м}} \hat{\theta}_j , \tag{4.67}$$

де $\hat{c}_i$ – оцінка стану *i*-ї стрілки з числа безлічі стрілок, що належать даному маршруту; ця оцінка розраховується по наступному принципу:

$$\hat{c}_i = \begin{cases} 0 - \text{якщо стрілка встановлена в положення, що не відповідає даному маршруту;} \\ . \\ 1 - \text{якщо стрілка встановлена в положення, що відповідає даному маршруту.} \\ . \end{cases} \tag{4.68}$$

$\hat{\theta}_j$ – оцінка стану *j-ї* ізольованої ділянки з числа безлічі ділянок, що належать даному маршруту; ця оцінка розраховується по наступному принципу:

$$\hat{\theta}_j = \begin{cases} 0 - \text{якщо ЗР даної ділянки знаходиться під струмом (тобто ділянка НЕ входить у зібраний маршрут);} \\ 1 - \text{якщо ЗР даної ділянки знеструмлене (тобто ділянка входить у зібраний маршрут).} \end{cases} \tag{4.69}$$

Відношення оцінки числа подій $\hat{K}_м$ до заданого для цього маршруту числу подій $K_м$ назвемо *ймовірністю маршруту* $P_м$:

$$P_м = \frac{\hat{K}_м}{K_м} . \tag{4.70}$$

Далі, із усіх M_{max} (кількість можливих маршрутів з однієї точки) отриманих таким способом ймовірностей відшукуємо максимальну. Маршрут, що відповідає цієї імовірності і буде, очевидно, зібраним маршрутом. Таким чином, визначення зібраного маршруту описується алгоритмом, що представлений на рис. 4.42.

ЦИКЛ по номеру маршруту *M*
    ЛІЧИЛЬНИК ПОДІЙ: = 0
    ОПИТУВАННЯ СТРІЛОК *M*-го маршруту
    ОПИТУВАННЯ ЗР *M*-го маршруту
    ЦИКЛ по кожній стрілці в *M*-му маршруті
        ЯКЩО положення стрілки відповідає маршруту ТО
            ЛІЧИЛЬНИК ПОДІЙ: = ЛІЧИЛЬНИК ПОДІЙ + 1
        КІНЕЦЬ ЦИКЛУ по кожній стрілці в *M*-му маршруті
    ЦИКЛ по кожній ділянці в *M*-му маршруті
        ЯКЩО ЗР ділянки знеструмлене ТО
            ЛІЧИЛЬНИК ПОДІЙ: = ЛІЧИЛЬНИК ПОДІЙ + 1
        КІНЕЦЬ ЦИКЛУ по кожній ділянці в *M*-му маршруті
    $\text{ЙМОВІРНІСТЬ МАРШРУТУ}(M) = \frac{\text{ЛІЧИЛЬНИК ПОДІЙ}}{\text{КІЛЬКІСТЬ ПОДІЙ}(M)}$
КІНЕЦЬ ЦИКЛУ по номеру маршруту *M*
ПОШУК МАКСИМАЛЬНОЇ ЙМОВІРНОСТІ МАРШРУТУ

Рис. 4.42. Алгоритм визначення зібраного маршруту

Даний алгоритм припускає, що для кожного з M_{max} маршрутів відомі ізольовані ділянки, а також усі стрілки і їхні положення, що відповідають зібраному маршруту. Також передбачається відомим кількість подій, що відповідають кожному зібраному маршруту з числа аналізованих.

4.9.4 Приклад реалізації алгоритму

Як приклад розглянемо ситуацію, коли при зібраному маршруті М3 (на рис. х.х він відзначений жирною лінією) перекручений сигнал від стрілки $З_{75}$. У табл. 4.15 показані лічильники подій і імовірності

ідентифікації кожного маршруту. Видно, що навіть при перекручуванні сигналу від стрілки $З_{75}$ зібраний маршрут М3 ідентифікований правильно.

Таблиця 4.15

Визначення імовірності ідентифікації зібраного маршруту

	Ділянки								Кількість подій що спостерігаються			Імовірність маршруту
	26		27		30		37					
	$З_{59}$	$З_{75}$	$З_{59}$	$З_{60}$	$З_{75}$	$З_{42}$	$З_{42}$	$З_{60}$	Стр.	ЗР	Σ	
Оцінка	1	1	1	1	1	0	0	1				
КМ1	–	×	×	×			×	–	0	2	2	2/5=0,4
КМ2	+	+					–	+	3	2	5	5/6=0,83
КМ3	+	–				×	+	+	3	3	6	6/7=0,86

перекруче ний сигнал

Примітка. Оцінки ділянок, для яких спостерігалася ЗР=0 (ділянка в зібраному маршруті) заштриховані.
"–" – оцінка не відповідає зібраній ділянці;
"+" – оцінка відповідає зібраній ділянці;
"×" – оцінка не грає ролі.

4.9.5 Ефективність запропонованого алгоритму

Аналіз статичного масиву $К_м$, підготовленого для парку відправлення станції Нижньодніпровськ-Вузол, показав, що при перекручування одного сигналу забезпечується 100% ймовірність визначення зібраного маршруту. При перекручуванні двох сигналів ймовірність визначення для різних ділянок складала від 30 до 70%. І лише при перекручуванні більш двох сигналів ймовірність визначення зібраного маршруту близька до нуля.

Зрозуміло, ймовірність визначення зібраного маршруту за даною методикою при відмовленні одного чи більшого числа сигналів залежить від конкретної топології станції.

Таким чином, можна зробити вивід, що розроблений алгоритм ідентифікації зібраних маршрутів, що використовує надлишкову інформацію від спеціальних (замикаючих) реле і заснований на принципі

максимальної правдоподібності, істотно підвищує ймовірність ідентифікації.

4.10 Синтез програмного забезпечення інформаційно-керуючих систем реального часу на основі теорії абстрактних автоматів

4.10.1. Проблеми автоматизації введення інформації в інформаційно-керуючі системи залізничного транспорту

Традиційно інформаційна підтримка роботи залізничних станцій реалізується потужними автоматизованими системами. В Україні, наприклад, для цих цілей використовується системи керування вантажними перевезеннями АСК ВП УЗ-Є [2]. Відомі подібні системи, в інших країнах Європи, США тощо [3,4]. Основне завдання подібних систем – збір даних, формування статистичної інформації, прогнозування, складання планів і т.п. Слабкою ланкою цих систем залишається відображення ситуації на станції в реальному масштабі часу. Для розв'язку такого завдання необхідно забезпечити автоматичне введення інформації про стан технологічного процесу на станції. Розв'язком таких завдань займалися вчені Дніпропетровського національного університету залізничного транспорту імені академіка В. Лазаряна [1]. Для введення інформації про поточний стан технологічного процесу можливе використання сигналів від наземного обладнання (рейкових кіл, стрілок, колійних датчиків і т.п.). Такі сигнали можуть приходити в інформаційну систему на різних фазах технологічного процесу. Правильна інтерпретація таких сигналів, зіставлення їх з конкретною фазою технологічного процесу є важливим і актуальним завданням при проектуванні алгоритмів роботи інформаційної системи [5].

Питаннями побудови імітаційних моделей роботи залізничних станцій на протязі тривалого часу займаються вчені багатьох країн, де

розвинене залізничне сполучення [6, 7, 8, 9]. Використовуються й різні механізми моделювання. Так у ряді робіт показане використання для моделювання роботи залізничних станцій мереж Петрі [10, 11, 12]. В інших роботах [13] автори використовують власні програмні засоби для побудови моделей технологічних процесів роботи станцій. В [7] розглядається цілий комплекс власних програмних моделей, що описують значну частину аспектів роботи станції. В [14] описана можливість використання для моделювання роботи станції автоматної моделі.

Імітаційні моделей використовуються для позасистемного аналізу роботи станції, побудови тренажерів оперативно-диспетчерського персоналу, вони дозволяють визначити раціональний хід технологічного процесу, дати пропозиції по поліпшенню колійного розвитку станцій.

Для побудови на залізничних станціях інформаційних систем, що працюють у реальному масштабі часу, потрібен особливий підхід до моделювання. В імітаційних моделях вхідні сигнали формуються штучно – програмними компонентами або оператором системи. У системах реального часу вхідні для моделі сигнали – реальні сигнали від наземного обладнання, від систем більш високого рівня, у деяких випадках – від оперативного персоналу станції.

Інформаційна система реального часу на станції відслідковує ряд технологічних процесів. Для кожного з них повинна бути своя модель. Але здається доречним розробка уніфікованої методики створення автоматної моделі, яка дозволяє відстежити в реальному масштабі часу кожний технологічний процес на станції, що володіє заданими властивостями. Більше того, ця модель повинна бути реалізована стандартним програмними компонентами. Для цього необхідно створити механізм проектування таких компонентів.

4.10.2 Проектування автоматної моделі

Ми будемо розглядати технологічні процеси, які мають чітко виражені дискретні стани, перехід між якими відбувається під впливом певних сигналів або при виникненні будь-яких подій. На залізничних станціях у такий спосіб можна представити більшу частину технологічних процесів. Наприклад, стан «огляд состава» ініціюється сигналом «включення огородження», а перехід до стану «завершення обробки состава» відбувається під впливом сигналу «вимкнення огородження». Контролер інформаційно-керуючої системи, пов'язаної з таким технологічним процесом (рис. 4.43), можна, при цьому, представити у вигляді класичного абстрактного автомата [15], тобто у вигляді шестикомпонентного вектора $\boldsymbol{S}=(\mathbf{A},\mathbf{Z},\mathbf{W},,,a_1)$, де:

$\mathbf{A} = \{a_1,a_2,,a_m\}$ – безліч станів (алфавіт станів);

$\mathbf{Z} = \{z_1,z_2,,z_n\}$ – безліч вхідних сигналів (вхідний алфавіт);

$\mathbf{W} = \{w_1,w_2,,w_t\}$ – безліч вихідних сигналів (вихідний алфавіт);

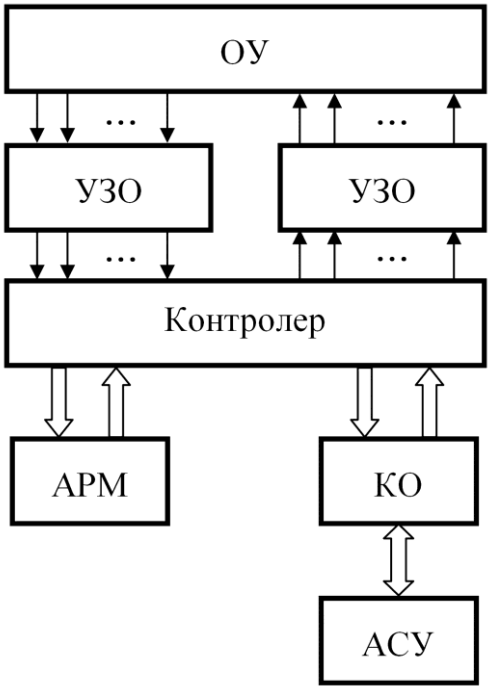
- функція переходів, яка парам стан-вхідний_сигнал (a_i,z_j) ставить у відповідність стан автомата: $a_s = (a_i,z_j)$, $a_s,a_i \ \mathbf{A}$, $z_j \ \mathbf{Z}$;

– функція виходів, яка парам стан-вхідний_сигнал (a_i,z_j) ставить у відповідність вихідні сигнали автомата: $w_q = (a_i,z_j)$, $w_q \ \mathbf{W}$;

$a_1 \ \mathbf{A}$ – початковий стан автомата.

У якості алфавіту станів можна представити безліч (кінцеву) станів технологічного процесу.

Безлічі вхідних сигналів (вхідному алфавіту) можна поставити у відповідність безліч вхідних сигналів і повідомлень, що надходять у контролер (сигнали від рейкових кіл, контролю положення стрілок, повідомлення від АСУ верхнього рівня й т.п.), а також безліч подій, що виникають у результаті комбінацій цих сигналів і повідомлень (наприклад, «відкрився вхідний світлофор + заняте вхідне рейкове коло + було повідомлення про прибуття поїзда з АСУ верхнього рівня»).



ОУ – об'єкт керування; ОЗО – обладнання зв'язку з об'єктом;
КО – комутаційне обладнання; АСУ – АСУ верхнього рівня.

Рис. 4.43. Контролер, що керує технологічним процесом

У якості вихідного алфавіту можна представить безліч дій, які виконує контролер у відповідь на вхідні сигнали (для інформаційної системи – це повідомлення, що пересилаються в АРМ і в АСУ верхнього рівня, а для керуючої системи – це ще й набір керуючих сигналів, що посилає контролер на виконавчі механізми).

На відміну від класичного представлення абстрактного автомата [15], який працює в дискретному часі (t = 0, 1, 2,) автомат, що представляє технологічний процес, працює в безперервному часі. Сигнали, що надходять на його вхід, приходять у випадкові моменти часу. Для приведення цього автомата до канонічного виду ці моменти часу можна представити у вигляді кінцевого дискретного нумерованого набору (масиву), тобто $t = t_0, t_1, t_2, .$

Завдання розроблювача – чітко визначити набір станів технологічного процесу, вхідний і вихідний алфавіт. Завдання ця

неоднозначне й успіх його багато в чому визначається мистецтвом розроблювача, знанням їм усіх тонкощів технологічного процесу.

У якості приклад наведемо (рис. 4.44) частину автоматної моделі інформаційної підсистеми парку прибуття, що є складовою частиною інформаційно-керуючої системи сортувальної станції [16].

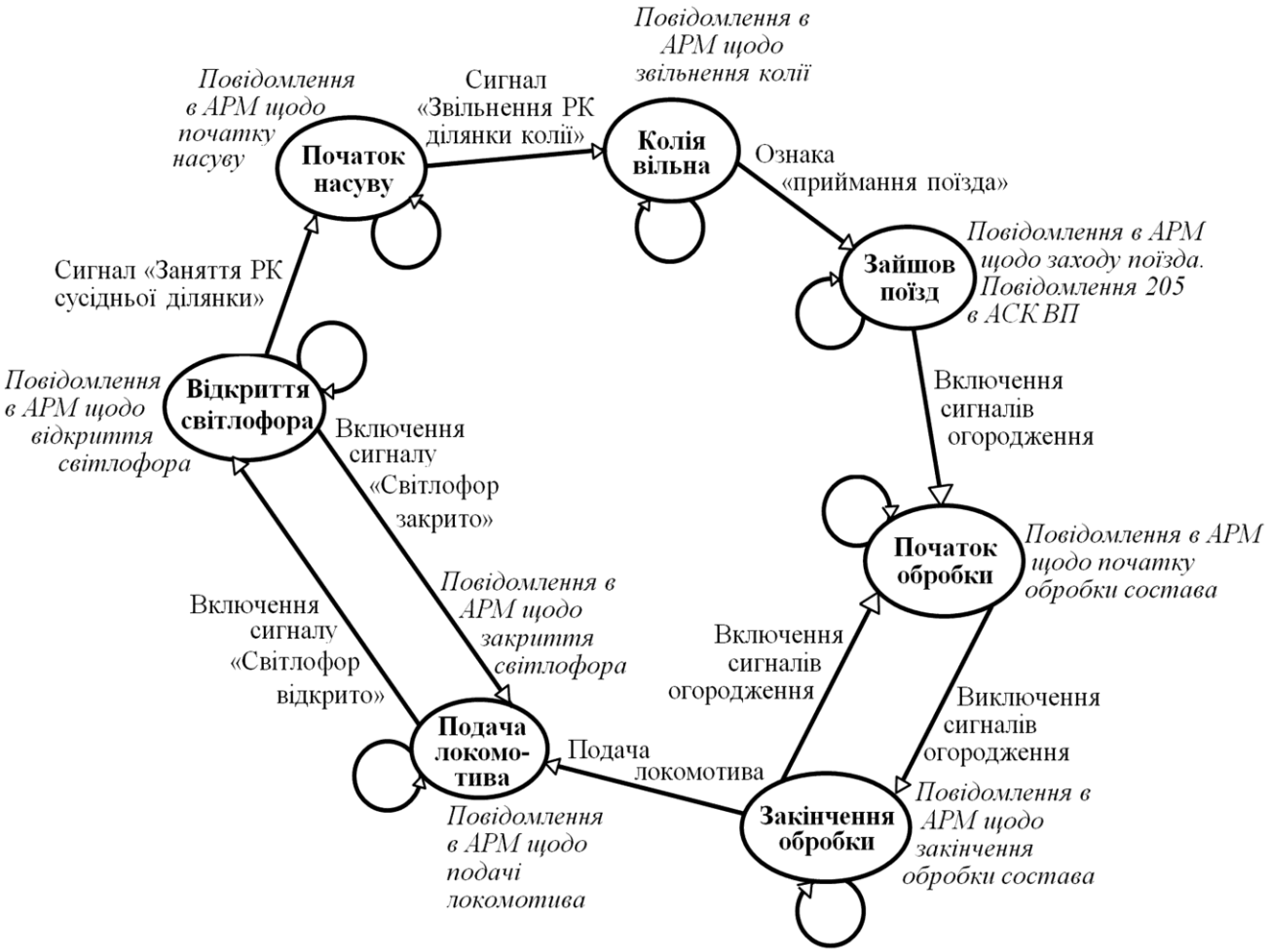


Рис. 4.44. Частина автоматної моделі інформаційної підсистеми парку прибуття сортувальної станції

У цьому прикладі технологічний процес обробки состава, що прибув на колію парку прибуття, представлений у вигляді асинхронного автомата Мілі, тобто вихідний сигнал визначається станом автомата й вхідним сигналом, який перевів автомат у даний стан.

Вершини графа представляють стани автомату, а точніше – стани технологічного процесу обробки поїзда в парку прибуття. На початку

стрілок, що вказують переходи з одного стану в інший, наведені сигнали, що ініціюють це переведення, а наприкінці стрілок, курсивом, зазначені ті дії (вихідні сигнали автомата), які виконуються інформаційно-керуючою системою при цьому переході.

Вихідний стан автомата – «Колія вільна». Ознака «Приймання поїзда» формується, як набір подій: перемикання сигналу вхідного світлофору з дозвільного на заборонний, сигнали положення стрілок визначають маршрут поїзда від головної колії на задану колію прибуття, включається сигнал звільнення рейкового кола ділянки перед заданою колією прибуття. Комбінація цих сигналів визначають перехід автомата в стан «Зайшов поїзд». Цей перехід ініціює формування спеціального повідомлення в АРМ чергового й в інформаційну систему вантажних перевезень про захід поїзда на станцію.

Після заходу поїзда на колію парку прибуття (і відчеплення локомотива) необхідно виконати обробку состава (технічний огляд, підготовка до розпуску). Дії бригади огляду починається після включення спеціальних сигналів огородження. Наявність цих сигналів визначає перехід автомата в стан «Початок обробки». Вимикання цих сигналів – перехід автомата в стан «Кінець обробки». Переходи супроводжуються формуванням в АРМі чергового відповідних повідомлень. У деяких випадках після відключення сигналів огородження потрібно провести додатковий огляд (перехід зі стану «Закінчення обробки» у стан «Початок обробки»).

По деяких комбінаціях сигналів рейкових кіл і маневрових світлофорів можна визначити факт подачі локомотива до составу, що був оброблений (ознака «Подача локомотива», перехід у стан «Подача локомотива», формування повідомлення в АРМ).

Відповідно до сигналів, показаних на рис. 4.44, відбуваються подальші переходи: «Відкриття світлофора» (сигнал від світлофора), «Початок насуву» (гірковий локомотив почав насувати склад на

сортувальну гірку, сформувався сигнал рейкового кола від ділянки, що примикає до колії прибуття, де стояв состав). Після звільнення ділянки, що примикає до колії ділянки прибуття (сигнал від рейкового кола) – перехід до стану «Колія вільна». Усі переходи супроводжуються формуванням і передачею повідомлень в АРМ чергового по парку та записуються у відповідну базу даних.

Передбачається, що для кожного з декількох колій парку прибуття є своя автоматна модель. Ці моделі повинні працювати паралельно.

Можна допустити, що існують технологічні процеси, які описуються асинхронним автоматом Мура, тобто їхній вихідний сигнал визначається тільки їх станом, незалежно від вхідного сигналу, який перевів їх у даний стан.

4.10.3 Синтез програмного продукту

У класичній теорії автоматів, абстрактне представлення необхідне для наступного структурного синтезу. Тому що ми розглядаємо заздалегідь відому технічну структуру (рис. 4.43), то наступним етапом після автоматного представлення технологічного процесу буде етап синтезу програмного продукту, що реалізує це представлення, для контролера.

Відомі ряд механізмів для синтезу програм мікроконтролерів [17]. У грудні 1993 року Міжнародна електротехнічна комісія (МЕК) визнала п'ять стандартних мов програмування, які можуть бути використані для реалізації або технологічних, або дискретних програмованих контролерів і їх загальні скорочення, такі як: сходова діаграма (LD), список інструкцій (IL), функціональна блок-схема (FBD), структурований текст (ST) і послідовна функціональна діаграма (SFC). Третє видання було опубліковано в лютому 2013 року.

Найбільш близьким до автоматного опису є послідовна функціональна діаграма (SFC). Послідовні діаграми функцій розбивають послідовне завдання на кроки, переходи й дії. Вони нарисовані графічно

для опису послідовності взаємодій [17]. В [18] коротко показане, як реалізувати діаграми з використанням мови програмування Ladder Diagram (LD).

Для керування технологічними процесами на залізничних станціях потрібні складні алгоритми й операційні системи реального часу. Однієї з найбільш досконалих систем реального часу є ОС QNX. Ця система містить власні засоби реалізації паралелізму. Це дозволяє використовувати опис технологічного процесу станції у вигляді набору окремих паралельно й незалежно працюючих кінцевих автоматів.

Для розробки систем реального часу в середовищі QNX часто використовується графічне середовище Photon application Builder (Phab). У цім середовищі для розробки програмного забезпечення пропонується мова С. Тому далі ми будемо розглядати можливість синтезу програмного продукту в середовищі Phab мовою С. Програмний продукт повинен реалізувати технологічний процес, представлений у вигляді автомата Мілі.

Розглянемо фрагмент асинхронного автомата Мілі (рис.4.45). Будемо вважати, що безліч вхідних сигналів, які не переводять автомат в інший стан, не формують вихідних сигналів.

Для кожного стану a_i визначимо безліч вхідних сигналів $\tilde{Z}_i \subseteq \mathbf{Z}$, $\tilde{Z}_i = \{z_{i1}, z_{i2}, \ldots, z_{ik_i}\}$, які переводять автомат зі стану a_i у будь-який інший стан $a_j\mathbf{A}$, відповідно до таблиці переходів, причому $a_j a_i$, а k_i – кількість таких сигналів (очевидно, що k_i n, де n – кількість сигналів у вхідному алфавіті).

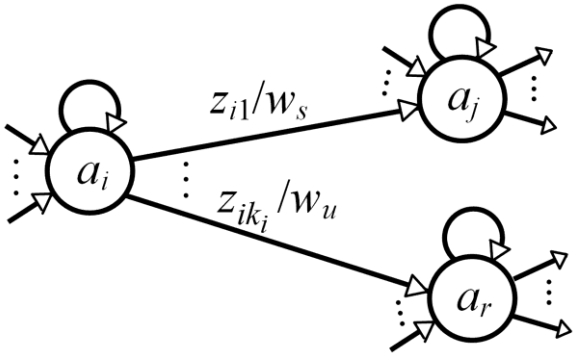


Рис. 4.45. Фрагмент асинхронного автомата Мілі

Можна запропонувати формальний алгоритм переходу від графа автомата (наприклад, рис. 4.45) до схеми алгоритму керуючої програми (рис 4.46), яка запускається при появі сигналу $z_{ij}\mathbf{Z}$ (нагадаємо, що під терміном «вхідний сигнал» ми домовилися розуміти як реальні сигнали, що надходять від наземного обладнання, так і повідомлення, що надходять від АСУ верхнього рівня, а також події, що виникають у результаті комбінацій цих сигналів і повідомлень).

Насамперед, аналізується поточний стан автомата (зі станом автомата асоціюється стан технологічного процесу, так, наприклад, як це показане на рис 4.44). Для кожного стану аналізується вхідний сигнал. Якщо для i-го стану прийшов вхідний сигнал $z_{i1}\tilde{Z}_i$, то виконується гілка алгоритму, у якій реалізуються дії, відповідні до вихідного сигналу w_s, а потім поточному стану привласнюється значення a_j (w_s, a_j визначаються по таблицях виходів і переходів автомата відповідно).

Таким чином, маючи в якості вихідних даних угоди про алфавіт станів, про вхідний і вихідний алфавіт, таблиці виходів і переходів автомата, а також безліч наборів $\tilde{Z}_i \subseteq \mathbf{Z}$ (i=1, 2, , m) можна за наведеними вище правилами формально скласти алгоритм реакції системи на появу будь-якого вхідного сигналу.

Блоку, у якому реалізуються дії, відповідні до вихідного сигналу w (рис. 4.46) можна поставити у відповідність виклик відповідної процедури. При цьому можна реалізувати універсальну програмну реалізацію вищезгаданого алгоритму. А настроювання під конкретний автомат виконується в модулі, де описуються вихідні дані автомата (таблиці), а також процедури, що реалізують дії, відповідні до вихідних сигналів.

Подібна методика [19] була використана при проектуванні програмного забезпечення для інформаційної системи сортувальної станції Нижньодніпровськ-Вузол. Дана система була розроблена в короткий

термін і успішно пройшла випробування на лабораторному полігоні університету.

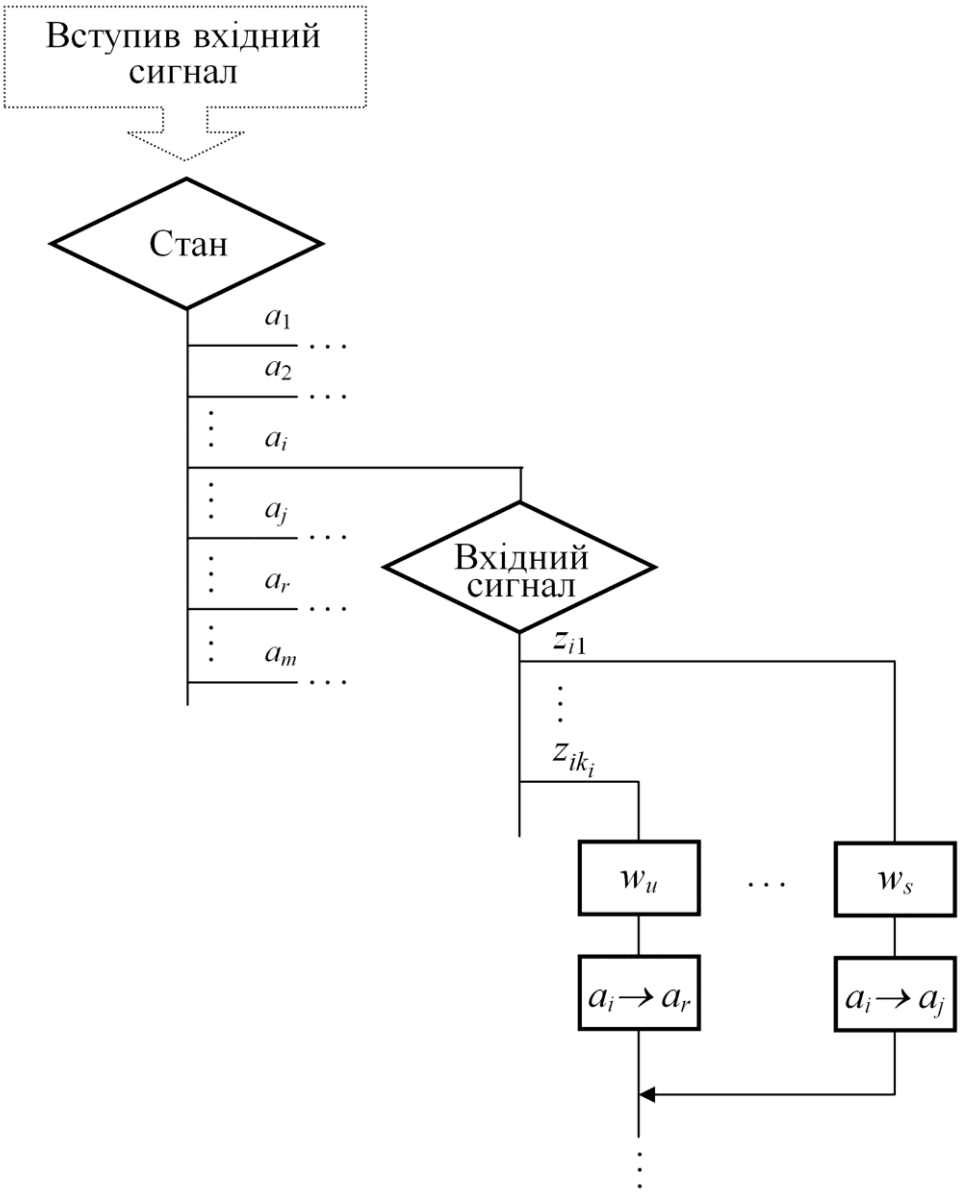


Рис. 4.46. Фрагмент схеми алгоритму реакції системи (відповідної до автомата рис. 4.45) на вхідний сигнал

Посилання у параграфі:

1. Шафит, Е.М., Жуковицкий И.В., Косорига Ю.А., Елисеев В.В. Совершенствование автоматизированных систем управления технологическими процессами на сортировочных станциях. *Информационно-управляющие системи на железнодорожном транспорте.* 2003. № 5. С. 53-55.
2. Косолапов А.А., Жуковицкий И.В. Резервы архитектуры автоматизированной системы управления грузовыми перевозками Украинских железных дорог. *Залізничний транспорт України.* 2013. № 1. С. 10-13.
3. Cherneva,G., Andonov, A.,Hubenov á, Z. The problem of synthesis of functionally steady information systems in railway transport. In: *5th International Scientific Conference "Theoretical and Practical Issues in Transport"* (11-12 February 2010, Pardubice, Czech Republic). Available at: http://hdl.handle.net/10195/37758
4. Константінов, Д., Мисько, Мю Дослідження перспективних систем диспетчерського управління на основі сучасних інформаційних технологій. *Збірник наукових праць УкрДАЗТ*, 2013, вип. 140 – С.91-96
5. Жуковицкий, И. Синтез программного обеспечения информационно-управляющих систем на основе теории абстрактных автоматов. *Інформаційно-керуючі системи на залізничному транспорті.* 2001. № 5. С. 39-41.
6. Лещинский, Е. *Имитационное моделирование на железнодорожном транспорте*. Москва: Транспорт. 1977. 176 с.
7. Вернигора, Р., Малашкин, В. Моделирование работы систем станционной автоматики в эргатических имитационных моделях железнодорожных станций. *Сб. науч. трудов Днепропетр. нац. ун-та ж.д. тр-та «Транспортные системы и технологии перевозок».* – 2011. № 2. С. 31-37.
8. Szűcs, G. Railway Simulation with the CASS NDRA Simulation System. *Journal of Computing and Information Technology.* 2001. Vol. 9. Nr 2. – P. 133–142.
9. Сукач, Е. Стенд имитационного моделирования сортировочной станции железнодорожной сети. *Проблемы программирования.* 2009. № 3. – С. 81-89.
10. Нагорный, Е. Алешинский, Е. Моделирование функционирования комплекса «Сортировочная станция – прилегающие участки» с помощью сетей Петри. *Информационно- управляющие системы на железнодорожном транспорте.* 2000. № 2. С. 98–103.
11. Giua, A., Seatzu, C. Modeling and supervisory control of railway networks using Petri nets. *Automation Science and Engineering, IEEE Transactions on.* 2008. Vol. 5. Nr 3. P. 431-445.

12. Milinkovic, S., Markovic, S., Veskovic, M. , et al. A fuzzy Petri net model to estimate train delays. *Simulation Modelling Practice and Theory*. 2013. Nr 33. P. 144-157.

13. Бобровский, В., Вернигора, Р. Функциональное моделирование железнодорожных станций в тренажерах оперативно-диспетчерского персонала. *Математическое моделирование*. 2000. № 2(5). С. 68-71.

14. Bobrovskiy, V. I., Kozachenko, D.N., Vernygora, R.V. Functional simulation of railway stations on the basis of finite-state automata. *Transport Problems*. 2014. Vol. 9. Issue 3. P. 57-66.

15. Баранов С.И. *Синтез микропрограммных автоматов* - Л.: Энергия, 1979. – 232 с.

16. Шафит, Е., Жуковицкий, И., Косолапов А., Туник Т. Выбор технических средств и разработка вариантов технической структуры СКАТ-СС. *Інформаційно-керуючі системи на залізничному транспорті.* 2000. № 4. С. 127.

17. Charles M. Fialkowski. Sequential function chart programming. Processing sequential and parallel operations based on time or events. Available at: https://www.isa.org/intech/20140804/

18. James McWhinnie. Sequential Function Charts for All. Implementing Sequential Function Chart Designs using Ladder Diagram Programming Language for Programmable Logic Controllers. Napier University, Edinburgh, Scotland. Available at:
http://www.plcdev.com/sequential_function_charts_all

19. Zhukovyts'kyy I. Use of an automaton model for the designing of real-time information systems in the railway stations / I. Zhukovyts'kyy // Transport problems. 2017. Vol. 12. No. 4. P.101–108. DOI: 10.20858/tp.2017.12.4.10 (Scopus)

РОЗДІЛ 5

КОМПЛЕКСНИЙ ПІДХІД ДО КОЕЦЕПТУАЛЬНОГО ПРОЕКТУВАННЯ І ВДОСКОНАЛЕННЯ ІНФОРМАЦІЙНИХ СИСТЕМ РЕАЛЬНОГО МАСШТАБУ ЧАСУу

5.1 Процеси проектування інформаційних систем

Розробка і впровадження сучасних інформаційних технологій і систем в усіх галузях промисловості і транспорту відбуваються в умовах глобальних структурних соціально-економічних, технологічних і соціальних змін. До основних чинників, що впливають на процеси інформатизації, а також на проектування автоматизованих систем та їх архітектуру можна віднести:

1 - **тенденції глобалізації** соціально-економічних відносин, що приводять до утворення транснаціональних корпорацій з територіально розподіленими підприємствами, єдиним центром управління з відповідною інформаційною інфраструктурою, потоками інформації і широким колом завдань автоматизації управління корпораціями;

2 – **інтеграція інформаційних систем** в економічних зонах, що створюються, з чіткою економічною незалежністю кожного учасника співпраці і прагненням до досягнення максимальної економічної ефективності спільної діяльності;

3 – територіальна розосередженість об'єктів управління і інформаційного забезпечення привели до широкого впровадження ІНТЕРНЕТ-технологій, які характеризуються **різноманіттям варіантів програмно-технічних рішень по організації доступу до технічних, програмних, інформаційних ресурсів корпоративних систем;**

4 – відкритість ІНТЕРНЕТ-технологій і широка розгалуженість глобальних комунікацій робить актуальним **завдання забезпечення**

надійності, живучості і інформаційної безпеки в процесі розробки корпоративних інформаційних систем;

5 - інтеграція задач вимірювання, контролю, управління і інформаційного забезпечення в сучасних системах привела до уніфікації методологій їх проектування, яка полягає в тому, що розглядаються загальні **моделі і методи проектування інформаційних систем (без виділення, як було раніше, вимірювальних автоматизованих систем, систем контролю, керуючих систем і, тим більше, з доповненням – обчислювальних систем);**

6 – інтеграція різних систем, технологій і задач управління, які розроблялися різними організаціями і розробниками в різний час виявила одну з щонайгостріших проблем інтеграції: **багатократне дублювання в інформаційному забезпеченні об'єднуваних систем (масивів, таблиць і інших інформаційних об'єктів), при їх часовій і семантичній неузгодженості; звідси – дублювання інформаційних потоків, «інформаційне засмічення» каналів і баз даних, а в результаті – проблема невірогідності даних і зниження ефективності управління;**

7 – всі вище перераховані чинники приводять до **експоненціального зростання складності** створюваних корпоративних інформаційних систем, що у свою чергу приводить до такого ускладнення процесів проектування, що вони переходять до класу «транс обчислювальних задач» [57], ефективно вирішувати які можна тільки із застосуванням моделей і методів штучного інтелекту.

W. L. Chapman, J. Rozenblit і A. T. Bahill довели [176], що системне проектування АСК високої складності, до яких відносяться АСК сортувальних станцій і гірок [69], є NP-повною задачею багато варіантного вибору ітераційного типу, пов'язаною з обробкою великого об'єму даних (див. рис. 5.1).

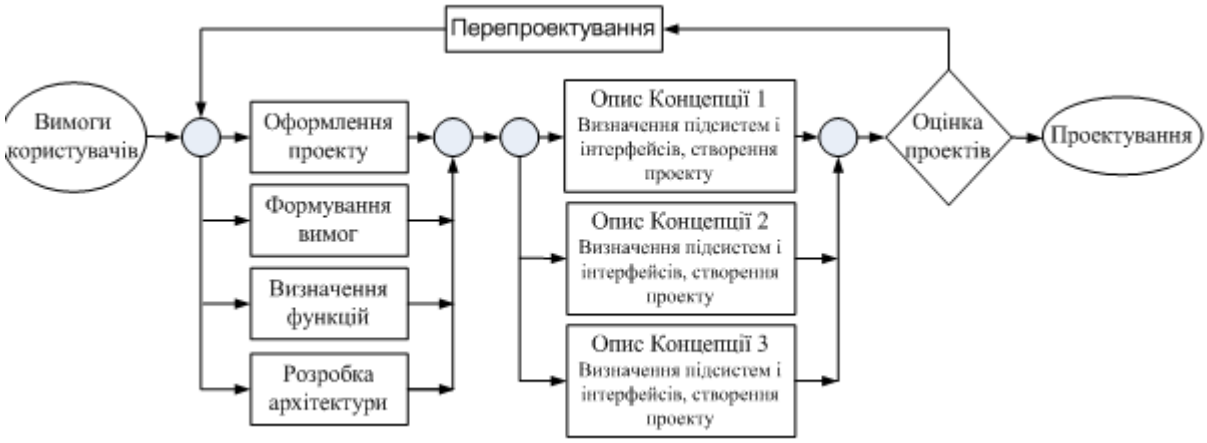


Рис. 5.1 Процес проектування систем автоматизації

Створення високоефективних інтегрованих автоматизованих систем керування, таких як АСК ВП УЗ-Є, які об'єднують функціонально і об'єктно-орієнтовані інформаційно-керуючі обчислювальні системи та мережі, практично неможливо без використання автоматизованих систем аналізу, проектування і навчання, які включають наукову методологію, програмні та технічні засоби проектування, і отримали назву інженерних технологій і інструментальних систем автоматизованого проектування. Це обумовлено зростанням складності нових програмно-технічних систем і високою ціною помилок, що допускаються на ранніх стадіях їх проектування. За оцінками експертів, виявлення та виправлення помилки на етапі проектування може коштувати в 2 рази, на етапі тестування - у 10 разів і на етапі експлуатації - в 100 разів дорожче, ніж на етапі аналізу [54].

Для залізничного транспорту в даний час, в умовах розробки інтелектуальної транспортної системи, актуальним є створення на базі ВУЗів єдиного наукового комплексу для рішення прикладних задач і наукового супроводження процесів автоматизації залізничного транспорту України [136].

5.2 Принципи і загальні вимоги до методології проектування систем реального масштабу часу

Розглянуті зміни в процесах інформатизації вимагають переосмислення і принципів проектування інформаційних систем [111]. Особливо систем реального масштабу часу, до яких відносяться АСК сортувальних станцій (див. п. 2.1).

За визначенням [203] "системою реального часу є така система, коректність функціонування якої визначається не тільки коректністю виконання обчислень, але і часом, в який отриманий необхідний результат. Якщо вимоги за часом не виконуються, то вважається, що відбулася відмова системи. Для того, щоб система могла задовольнити вимогам, що пред'являються до систем реального часу, апаратні, програмні засоби і алгоритми роботи системи повинні гарантувати задані часові параметри реакції системи. Час реакції не обов'язково повинен бути дуже маленьким, але він повинен бути гарантованим (що відповідає поставленим вимогам)". Тобто основні вимоги до систем реального часу можна графічно відобразити таким чином (див. рис. 5.2) [187].

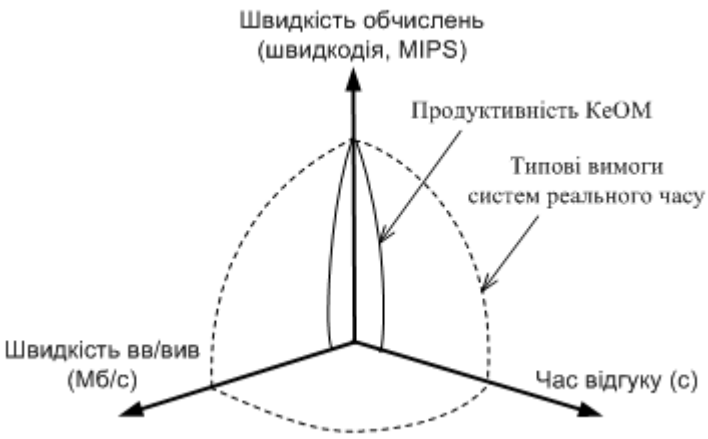


Рис. 5.2 Простір вимог для систем реального часу

Відзначимо головні принципи проектування систем:

S - принцип **системної** єдності і розвитку (вимагає використання відкритих моделей і методів проектування, стандартних протоколів і моделей взаємозв'язку відкритих систем);

D - принцип **декомпозиції** систем, що проектуються;

I - принцип **ієрархічності**;

M - принцип **багатоетапності** процесу проектування;

К - багато критеріальність проектування **з перевагою часових критеріїв** (див. п. 2.2.2);

P - принцип використання методології **паралельного** проектування;

R - принцип ітеративності (**рекурсивності**) проектування;

T - принцип **типізації -** уніфікації проектних рішень і проектних процедур;

A - принцип **автоматизації** процесів проектування і формування інформаційної бази проектних рішень.

Методологія проектування інформаційних систем повинна враховувати зміни, що відбуваються. По-перше, для складних інформаційних систем на перший план виходить завдання **багаторівневої структурної оптимізації**, яка повинна вирішуватися на декількох рівнях: рівні інфраструктури об'єктів автоматизації (топології розміщення об'єктів), інформаційної і технічної структури (див. розділ 3). По-друге, як уніфікована одиниця об'єму роботи, що виконується в системі, пропонується використовувати «φ-**транзакцію»** як розгалужену послідовність виконуємих системних і прикладних програм від моменту надходження заявки в систему до моменту завершення її обробки, пов'язаного або із занесенням інформації в базу даних, або з видачею керуючої дії, або відображенням інформації для персоналу системи (див. розділ 4).

Методика проектування складних систем повинна спиратися на

комплекс математичних і імітаційних моделей. Імітаційне моделювання застосовується для вирішення складних завдань оцінки режимів функціонування розподілених систем. Ці ж моделі можуть потім використовуватися для налагодження програмного забезпечення системи.

Методика повинна спиратися на засоби автоматизації виконання основних і найбільш трудомістких проектних процедур з формуванням бази проектних рішень у відповідній проблемній області.

5.3 Методики концептуального (системного) проектування складних ІС

Сучасні засоби автоматизованого проектування засновані на різних технологіях, методиках і схемах проектування, багато з яких доведені до рівня державних та міжнародних стандартів [194, 206, 165, 164, 209, 196].

Найбільше поширення при створенні складних систем отримав метод функціонального структурного аналізу і проектування Д. Росса SADT (Structural Analysis and Design Technique). Основні положення методу SADT, сформульовані майже 20 років тому, отримали розвиток в США в рамках проекту ICAM (Integrated Computer-Aided Manufacturing) по створенню інтегрованих АСК на базі комп'ютерних технологій. У проекті ICAM було розроблено сімейство методологій IDEF (ICAM DEFinition):

• IDEF0 - методологія створення функціональної моделі виробничого середовища або системи (заснована на методі SADT);

• IDEF1 - методологія створення інформаційної моделі виробничого середовища або системи (заснована на реляційній теорії Кодда і використанні ER - діаграм Чена);

• IDEF2 - методологія створення динамічної моделі виробничого середовища або системи.

Пізніше у складі проекту ICAM почали розробку проекту IISS (Integrated Information System Support) по створенню технології логічного і фізичного об'єднання неоднорідних обчислювальних систем в

обчислювальну мережу. У результаті цієї роботи з'явилася методологія семантичного моделювання даних IDEF1X як розширення методології IDEF1.

Інструментальна підтримка методології IDEF здійснюється за допомогою пакету IDEF/Design, розробленого фірмою Meta Software Corporation (США) [78].

Іншим напрямком розвитку методології SADT було створення у Великобританії промислової інформаційної технології SSADM (Structured Systems Analysis and Design Method), яка з 1981 року оголошена відкритим галузевим стандартом, обов'язковим для використання в усіх державних проектах автоматизованих систем (АС). У 1990 році була прийнята четверта версія технології SSADM, орієнтована на застосування інструментальних програмних засобів проектування. З середини 1980-х років працює Асоціація користувачів SSADM - International SSADM Users Group, що об'єднує понад 300 організацій, які використовують SSADM при створенні АС. У 1993 році технологія SSADM була офіційно прийнята в якості державного стандарту Великобританії. Ця методологія отримала визнання в багатьох країнах Західної Європи.

Методичне забезпечення технології SSADM включає тринадцять взаємопов'язаних методик проектування АС: визначення вимог до АС; моделювання інформаційних потоків; логічне моделювання даних; визначення функцій і завдань; динамічне моделювання даних; реляційний аналіз даних; вибір варіантів автоматизації; розробка демонстраційного прототипу; вибір варіантів технічної реалізації; проектування діалогової взаємодії; логічне проектування процедур обробки даних; фізичне проектування баз даних; фізичне проектування процедур обробки даних.

Перераховані методики різняться за ступенем формалізації і можливості автоматизації проектних процедур, які в них містяться. Найменш формалізованими є методики, які відносяться до ранніх стадій проектування систем. Тим не менш, в технології SSADM всі методики

мають чітко структуровані проектні процедури і формалізовані критерії оцінки якості результатів.

Програмні засоби автоматизації, що підтримують різні методики SSADM, розробляють кілька десятків фірм у Великобританії, проте, пройшли успішну сертифікацію тільки дев'ять програмних продуктів. Це говорить про велику наукоємність та програмну складність всіх інструментальних засобів автоматизації технологій проектування систем. Тому їх вартість, при невеликому тиражі, що не перевищує ста, досить висока і коливається від кількох тисяч доларів.

Особливістю американської (IDEF) та європейської (SSADM) технологій проектування є те, що вони, по-перше, охоплюють в основному питання проектування інформаційного і програмного забезпечення АС, і, по-друге, приділяють велику увагу управлінню розробкою та контролю якості проектування. Друга особливість виявляється в тому, що в технології SSADM, наприклад, проектна документація розділена на три категорії: технічна, організаційно-розпорядча і контролю якості. При цьому визначено конкретні вимоги до структури, змісту і критеріїв оцінки кожного документа. Окремі технології проектування регламентують тільки процес організації корпоративної розробки складних АС.

На залізничному транспорті до таких технологій можна віднести практичну методику PROMETEO (PROcedure, METhod and Organisation for international computer projects), розроблену міжнародним союзом залізниць UIC (International Union of Railways) для виконання проектів інформаційних обчислювальних систем, які охоплюють кілька взаємопов'язаних залізниць. Вона заснована на наступних методологічних принципах [161]: модульність проектів; чітке визначення кожної стадії: результати; дані для наступної стадії; наявність контрольних точок для перевірки «Хто? Що? Як? »; структурний зв'язок між різними рівнями контролю проекту; комп'ютерне застосування методів; "Процедура + Метод + Організація" = практичний довідник виконавця.

Методика PROMETEO передбачає розбиття проекту на фази, а кожної

фази - на стадії. Послідовність фаз і стадій розробки проектів за даною технологією представлена в таблиці 5.1 і на рисунку 5.2.

Розглянуті технології, які були обрані лише з причин своєї поширеності та комплексності охоплення проектно-дослідницьких робіт, являють собою лише найбільш типову, стандартизовану частину великої кількості інших моделей, методів і технологій проектування [153, 59].

У вітчизняній практиці проектування АС найбільш близькою за своїм призначенням, змістом та обсягом є інформаційна технологія проектування автоматизованих систем, описана в ГОСТ групи 34 [7]. На рисунку 2.11 представлена схема життєвого циклу складної АС, описана в термінах чотирьох технологій: ГОСТ 34, SSADM і PROMETEO.

Аналіз представлених технологій показує, що:

а) послідовна (каскадна) схема проектування [7] не відповідає рекурсивному характеру процесу дослідження і розробки складних систем; сучасні технології орієнтуються на ітераційність [56] і рекурсивність [153] проектування з паралельним вирішенням проектно-дослідницьких завдань;

б) технології проектування засновані на чіткій структуризації процесу створення складних систем, в якому для кожної стадії (фази, етапу) проектування визначається не тільки ЩО необхідно отримати [7, 163], але і ЯК це зробити [56];

в) у питанні ЯК вирішувати проектно-дослідницькі завдання при створенні складних систем всі підходи декларують - з використанням ЕОМ, але тільки IDEF і SSADM з розглянутих технологій, включають інструментальні засоби і докладні методики їх застосування;

г) найменш формалізованими, але найбільш важливим з позицій зменшення «дорогих помилок» у проекті, є ранні етапи аналізу і проектування систем;

д) більшість технологій та інструментальних засобів проектування призначені для рішення тільки окремих завдань: розробки інформаційного та програмного забезпечення, компонування технічної структури системи, управління проектами;

Таблиця 5.1

Методика PROMETEO

Фаза проектування		Стадії проектування	
Позначення	Найменування	Позначення	Найменування
P	Підготовка	P1	Представлення системи
		P2	Планування досліджень
M	Формування групи партнерів	M3	Формування групи партнерів
D	Проектування	D4	Функціональні дослідження
		D5	Архітектурне дослідження
I	Реалізація	I6	Залучення засобів
		I7	Реалізація
		I8	Приймальні випробування і затвердження
		I9	Реалізація систем партнерів
		I10	Підготовка до введення в дію
		I11	Введення в дію
O	Експлуатація	O12	Експлуатація

е) в якості моделей в методологіях проектування систем використовуються функціональні структурні моделі, інформаційні статичні і динамічні моделі (моделі станів, моделі процесів); для їх опису широко використовуються графічні схеми і табличні форми [54, 56, 153].

ж) новітні технології проектування (SDLC, E2AF, MDA, RM-ODP, RUP, TOGAF, логічна структура Захмана тощо [194]) більше уваги приділяють архітектурі систем керування підприємствами і так званим фреймворкам - чітким структурним і динамічним моделям багаторазового використання, в якій системи можуть бути описані і розроблені. Тобто фреймворк - це середовище, яке містить математичні моделі, методи, алгоритми, інструментальні програми, об'єднані в єдиній методиці дослідження, проектування і удосконалення систем.

Рис. 5.2 . Схеми життєвого циклу. Стандарти розробки автоматизованих систем

з) серед відомих проаналізованих методик відсутні методики проектування складних АСК на залізничному транспорті, які працюють в реальному масштабі часу.

5.4 Науково-методичний комплекс системного інтегратора (КСІ)

Науково-методичний комплекс системного інтегратора КСІ - це набір математичних моделей, методів, програмних інструментальних засобів, об'єднаних в інженерну методику системного аналізу, проектування і вдосконалення інформаційно-керуючих систем реального масштабу часу сортувальних станцій.

Ця методика є розвитком і узагальненням науково-методичного підходу до проектування складних систем, який розробляється на кафедрі ЕОМ Дніпропетровського національного університету залізничного транспорту імені академіка В. Лазаряна вже впродовж десятків років. Він знайшов застосування при розробці таких систем як АСКРСГ (ст. Ясинувата), АСКМД (ст. Пермь-Сортувальна, Росія), ІККСС (ст. Нижньодніпровськ-Вузол), ІПС СС (ст. Орєхово-Зуєво, Росія), система аналітичних серверів АСК ВП УЗ, корпоративна інформаційна мережа університету (ДНУЗТ) та інших інформаційно-керуючих систем [59, 108, 60, 115].

У розробленій методиці враховані сучасні принципи і вимоги до технологій створення складних інформаційних систем. Ця методика застосовується ітеративно при виконанні послідовності стадій, які відносяться до системного проектування і включають (відповідно до [7]): стадії формування вимог до системи, розробка концепції побудови автоматизованої системи, технічне завдання, ескізний і технічний проект (див. рис. 5.2).

Критерії оптимізації систем реального масштабу часу поділяють згідно з [187] на якісні ексклюзивні критерії (Х), якісні критерії поетапного

використання (G) і кількісні критерії (Q) (див. рис. 5.3).

Домінуючими критеріями і обмеженнями в процесі проектування АСК реального масштабу часу є часові якісні (X0, G0) і кількісні (Q1, Q1.1, Q1.2, Q1.3, Q1.4) характеристики. Вони доповнюються критеріями Q2 (Capacity reserves) щодо забезпечення необхідних резервів продуктивності, пропускної здатності, ємності (див. рис. 5.3) і, звичайно, Q3 - повні проектні вартості (верхні витрати - “the bottom line”).

Стандартом ISO число кількісних критеріїв обмежено двома метриками: час реакції (тривалість відповіді) і число транзакцій, які потрібно обробити в період часу [209].

Розглянуті критерії і покладені в розроблену методику, макросхема якої представлена на рисунку 5.4.

Концептуальну основу методології складають наступні положення.

1. Процес аналізу і проектування складних систем представляється у вигляді послідовно-ітераційної схеми поетапного пошуку раціональних проектних рішень з використанням евристичних методів оптимізації, яка може бути налаштована або пов'язана з послідовністю ранніх стадій, етапів, завдань інформаційної технології проектування АС за ГОСТ 34. При цьому до ранніх стадій віднесені стадії 1, 2, 3, 4, 5 (див.рис.5.2).

2. База знань методології представляється у вигляді набору онтологій сортувальних станцій, сортувальних гірок і комплексу програмно-технічних засобів промислових комп'ютерів Advantech [158, 84], на які зорієнтовані розробки систем автоматизації сортувальних станцій в Германії, Росії та Україні [69].

3. Методологія є адаптивною до набору початкових даних, наявних у розробника при аналізі і проектуванні конкретної системи. Це означає, що в базі знань активізується та послідовність проектних етапів і процедур, для яких є всі необхідні дані (або в базі знань, або введені проектувальником в режимі діалогу).

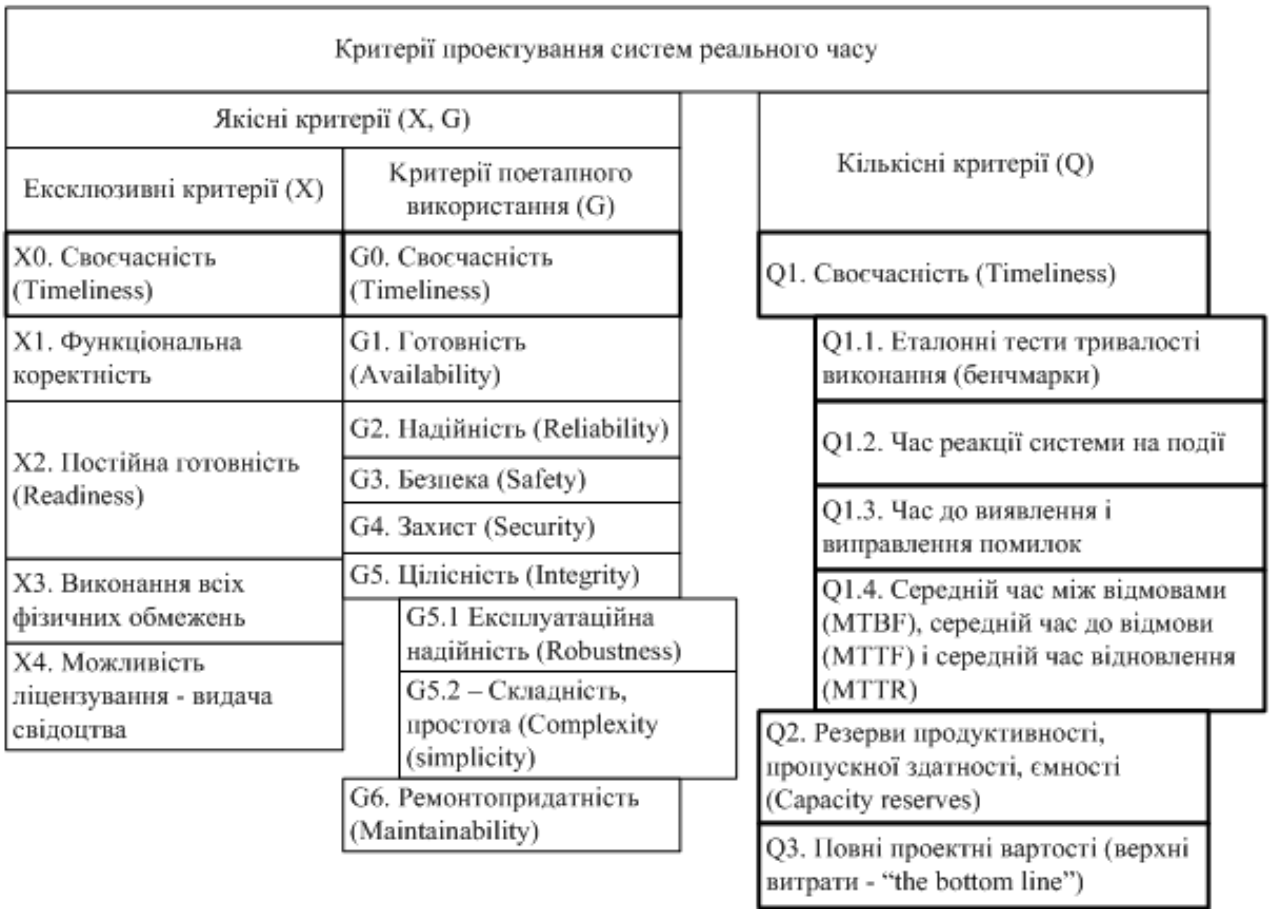


Рис. 5.3 Критерії оптимізації систем реального часу

4. Реалізація методології представляється у вигляді програми CSI комплексу системного інтегратора, програми CSProject для побудови і розрахунку інформаційно-часових характеристик φ-транзакцій комп'ютерних систем реального масштабу часу (п. 4.2), програми OPTiFLOW розподілу інформаційних потоків в системах реального часу (п. 3.1, 3.2), програми ОПТІКОС оптимізації інформаційно-керуючих систем (п. 3.3.1), програми GAOSIS генетичного алгоритму оптимізації структур інформаційних систем (п. 3.3.2) і програми PRIORY вибору пріоритетів потоків заявок в інформаційних системах (п. 4.3).

5. Вихідні дані і результати системного аналізу і проектування видаються в таблично-графічних формах.

Початковими даними для створення системи є:

блок 1 - характеристика об'єкта автоматизації: опис організаційно-технологічних характеристик системи, що проектується, який включає

технологічні підсистеми, їх структури, ділянки, потоки і характеристики сигналів на ділянках, засоби механізації і обладнання низової автоматики, топологія їх розміщення, приміщення на станції, показники надійності пристроїв, послідовність технологічних операцій, що виконує оперативний персонал системи, що проектується (потоки, місце, час); периферійне обладнання, яке повинно обслуговуватись АСК;

блок 2 - характеристика функціонування АСК (с позицій замовника): функції, задачі, алгоритми або програми, які має виконувати система, що проектується; для нових систем цей опис буде на рівні функцій і задач; для систем, що вдосконалюються або модернізуються - це розроблені алгоритми або програми з оцінкою частотного складу операцій (команд); для всіх рівнів з різним ступенем повноти і точності задаються компоненти інформаційного забезпечення (масиви); для кожного сигналу (заявки) вказуються керуючі дії або повідомлення персоналу, що видаються; опис оформлюється у вигляді φ-транзакцій в діалозі з програмою CSI;

блок 3 - опис технічних засобів автоматизації: керуючі ЕОМ, мікроконтролери, засоби вводу-виводу, нормалізації і комутації сигналів, пристрої живлення, конструктиви, мережеве обладнання, програмне забезпечення, показники надійності і вартості пристроїв;

блоки 4, 5, 6, 7 - описують критерії та обмеження бізнесові (Q2, Q3 - показники ресурсозбереження, витрати, втрати, вартість системи, показники надійності, які необхідно забезпечити), технологічні (елементи топології, канали), функціональні (Q1, Q2) і технічні (завантаженість процесорів, каналів, вартість програмно-технічних засобів, уніфікація рішень);

Проектування системи починається за трьома взаємопов'язаними напрямками. Перший - блоки 8, 9, 10, 11: оптимізація структури комунікацій на об'єкті автоматизації за критерієм мінімізації сумарної довжини комунікацій з фіксованими каналами (побудова мінімального кістякового дерева). Другий - блоки 12, 13, 14, 15: оптимізація

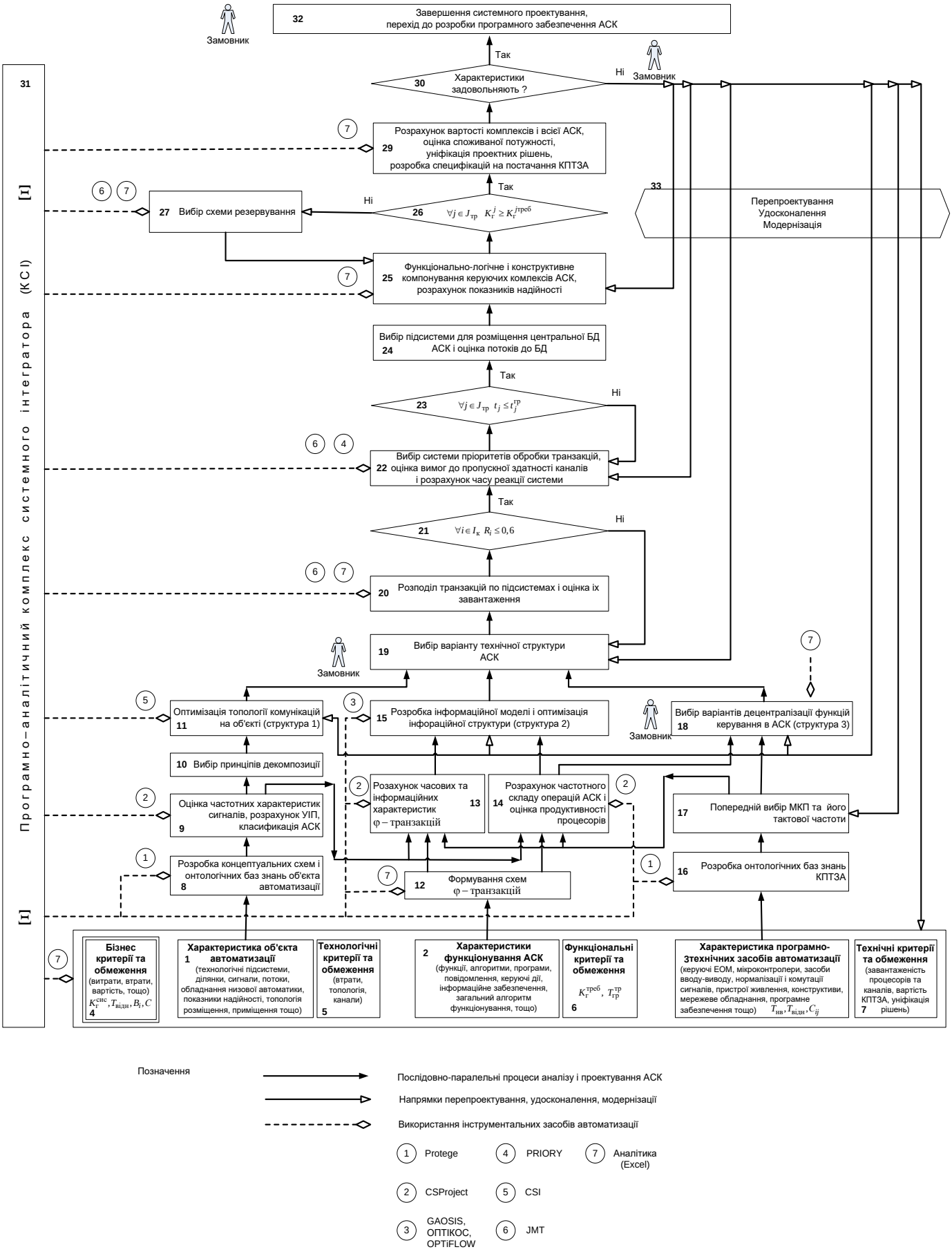


Рис. 5.4 Науково-методичний комплекс системного інтегратора - (КСІ)

інформаційної структури АСК за критерієм мінімізації приросту сумарного інформаційного потоку при зменшенні кількості інформаційних зв'язків в структурі (див. п. 3.3, 3.4). Третій напрямок - блоки 16,17,18: вибір варіантів децентралізації функцій в ієрархічній структурі за критеріями мінімізації сумарних втрат або вартості систем (див. п. 3.4).

В рамках першого і третього напрямків формуються відповідно онтології сортувальної станції і гірки (див. п. 2.2.2) і комплексу програмно-технічних засобів [84].

Перша онтологія використовується в блоках 9, 10, 11, 19, 24, 25, 29.

Друга онтологія - в блоках 17, 18, 24, 25, 29.

На основі сформованих трьох варіантів структури обирається за участю замовника, який представляє поточні бізнес-інтереси (пріоритети), варіант технічної структури (блок 19) для попередньо обраного типу мікропроцесора (блок 17).

На другому напрямку на основі початкових даних (блок 2) формуються φ-транзакції (блок 12), що описані в розділі 4, і розраховуються їх часові та інформаційні характеристики (блок 13) для оптимізації інформаційної структури (блок 15). Крім того, розраховуються частотний склад операцій, що використовуються в системі, і продуктивність процесорів в метриках MIPS і в φ-транзакціях/с для пошуку оптимальних ієрархічних структур (блок 18).

Для обраного варіанта технічної структури φ-транзакції розподіляються по підсистемах і оцінюється їх завантаження (блоки 20, 21). Якщо це обмеження виконується, для розподіленої АСК для кожної φ-транзакції розраховується час реакції системи, обирається оптимальний пріоритет (п. 4.3) і вираховуються вимоги до пропускної здатності каналів (блок 22). Далі, при виконанні часових обмежень (блок 23), на основі даних блока 13, вибирається підсистема для розміщення центральної бази даних в децентралізованій АСК (блок 24).

На наступному етапі (блок 25) для кожної підсистеми виконується

функціонально-логічне і конструктивне компонування відповідного комп'ютерного комплексу з використанням онтології КПТЗА (блок 16). Крім цього обирається технологія і засоби зв'язку в системі з тієї ж онтології. Ці засоби повинні відповідати вимогам з пропускної здатності каналів (блок 22). Для отриманої технічної структури виконується оцінка коефіцієнтів готовності і, якщо вони не забезпечується (блок 26), вибирається схема резервування слабкого елемента (блок 27).

Якщо потрібна надійність забезпечена, то для АСК виконується уніфікація проектних програмно-технічних рішень і складаються специфікації, розраховуються вартості комплексів і всієї системи, споживана потужність і експлуатаційні показники системи (блок 29). Результати проектування розглядає замовник (блок 30) і, якщо вони його улаштовують, системне проектування завершується і починається розробка програмного забезпечення (блок 32). Якщо "Ні", вибирається напрямок перепроектування (блок 32). З цього ж місця починається і вдосконалення або модернізація системи.

На рисунку 5.4 вказані інструментальні засоби, які використовуються в процесі дослідження и проектування АСК.

Розроблений науково-методичний комплекс системного інтегратора (КСІ) використовується не лише для системного проектування і вдосконалення систем автоматизації сортувальних станцій, але і в навчальному процесі при підготовці бакалаврів і магістрів з комп’ютерних систем та мережі напряму «Комп’ютерна інженерія» в Дніпроському національному університеті залізничного транспорту імені академіка В. Лазаряна, а також в курсовому і дипломному проектуванні.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Автоматизированная система организационно-технологического управления сортировочными станциями (АСОТУ СС). Разработка технического задания на ситему в целом [Текст] : Звіт про НДР. Т. 99.01.85.87 / Дніпропетровський інститут інженерів залізничного транспорту; керівн. А.А. Косолапов; викон. : А.А. Косолапов. — Дніпропетровськ : ДІІТ, 1986. — 95 с. - держреєстрація № 01860063785.

2. Всероссийская практическая конференция «Стандарты в проектах современных информационных систем» [Электронный ресурс] / Режим доступа : http://www.osp.ru/cw/2003/20/64757/ // Computerworld Россия. 2003. № 20.

3. ГОСТ 24.701-83 Автоматизированные системы управления техно-логическими процессами. Надежность. Основные понятия. [Текст]. — М : Издательство стандартов, 1983. — 17 с.

4. ГОСТ 34.003-90 [Текст] // Информационная технология. Комплекс стандартов на автоматизированные системы. Автоматизированные системы. Термины и определения. 1990.

5. ГОСТ 23501.108-85 [Текст] // Системы автоматизированного проектирования. Классификация и обозначение. 1985.

6. Замедлитель вагонный РНЗ-2М [Електронний ресурс] / Режим доступу : http://www.zrmz.ru/catalog/24/88/ // Златоустовский ремонтно-механический завод. 2013.

7. Информационная технология. Комплекс стандартов и руководящих документов на автоматизированные системы (ГОСТ 34.201-89, ГОСТ 34.602-89, РД-50-682-89, РД-50-680-88, ГОСТ 34.601-90, ГОСТ 34.401-90, РД-50-698-90, ГОСТ 34.003-90, РД-50-34.119-90) [Текст] —М: Госстандарт СССР, 1991. — 143 с.

8. Концепція автоматизації роботи сортувальних гірок України [Текст] / Проект. : ДП ДНДЦУЗ, 2010. — 36 с.

9. HIPO-технология. Общее описание. [Текст]. — М : ПКТБ АСУЖТ, 1980. — 48 с.

10. Науково-дослідна та дослідно-конструкторська робота [Электронный ресурс] // Режим доступа: URL: http://www.youtube.com/watch?v=3gjrlBp9rvc&feature=relmfu/. 2013.

11. "НТЦ ТРАНГССИСТЕМОТЕХНИКА": на высоте сегодняшних требований [Текст] // Евразия Вести. 2011. № II.

12. Общеотраслевые руководящие методические материалы по созданию и применению автоматизированных систем управления технологическими процессами в отраслях промышленности (ОРММ-3 АСУТП) [Текст] / Государственный комитет СССР по науке и технике. — М : Финансы и статистика, 1986. — 73 р.

13. Про схвалення Транспортної стратегії України на період до 2020 року // Розпорядження Кабінету Міністрів України від 20 жовтня 2010 р. № 2174-р, Київ. 2010.

14. Рекомендации по механизации и автоматизации сортировки вагонов на горках. Р 835 [Текст] // III издание. Разработано экспертами Комиссии ОСЖД по инфраструктуре и подвижному составу 25-27 мая 2004 г. — Варшава, 2004. — 11 с.

15. Техническое задание на создание автоматизированной системы управления маршрутами движения отцепов на сортировочной горке ГАЦ (АСУМД) [Текст] —М: МПС, 1981. — 40 с.

16. "Укрзализныця" запустила Единый центр обработки данных / Интерфакс-Украина, 26.07.2012. // [Электронный ресурс]. - Режим доступа: URL: http://interfax.com.ua/news/economic/112240.html 2012.

17. Ададуров С.Е. Интеллектуальный железнодорожный транспорт [Текст] // Автоматика, связь, информатика. 2011. № 6. — С. 4-8.

18. Алиев Т.И. Основы моделирования дискретных систем [Текст] / Т.И. Алиев. — Санкт-Петербург : СПбГУ ИТМО, 2009. — 363 с.

19. Архангельский Е.В. Надёжность технических устройств и её влияние на перерабатывающую способность станций [Текст] / Е.В. Архангельский, В.П. Шейкин // Вестник ВНИИЖТ. 1979. № 7. — С. 1-5.

20. АС № 45271 від 21.08.2012, Україна, ДДІВ. Комп'ютерна програма "Генетичний алгоритм оптимізації структур інформаційних систем (GAOSIS)" / Косолапов А.А., Гриненко Ю.О. . 2012.

21. АС № 45855 від 02.10.2012, Україна, ДДІВ. Комп'ютерна програма "Вибір пріоритетів потоків заявок в інформаційно-керуючих системах реального часу» («PRIORY»)" / Косолапов А.А., Борисов О.І., 2012.

22. АС № 45857 від 02.10.2012, Україна, ДДІВ. Комп'ютерна програма "Оптимізація інформаційно-керуючих обчислювальних систем» («ОПТІКОС»)" / Косолапов А.А., Мудрик О.Б., 2012.

23. АС № 45858 від 02.10.2012, Україна, ДДІВ. Комп'ютерна програма "Комплекс системного інтегратора» («CSI»)" / Косолапов А.А., Магамедов В.К., 2012.

24. АС № 1073146, МКИ, В61, L17/00. Устройство для управления технологическим процессом роспуска составов на сортировочной горке / Г.А. Красовский, Л.В. Сафрис, А.А. Косолапов, Б.М. Филимонов, Е.М. Шафит, Е.В. Щербаков. - № 3294504/27-11; заявлено 03.04.81; опубл. 15.02.84, бюл. №6. — СССР, 1984. — 2 с..

25. Асанов М.О. Дискретная математика: графы, матроиды, алгоритмы [Текст] / М.О. Асанов, В.А. Барановский, В.В. Расин —Ижевск : НИЦ "Регулярная и хаотическая динамика", 2001. — 288 р.

26. Байцер Б. Архитектура вычислительных систем [Текст] / Б. Байцер. Т. 1 — Москва : МИР, 1974. — 498 р.

27. Беляев Ю.К. Надежность технических систем : Справочник. Под. ред. проф. И.А. Ушакова [Текст] / Ю.К. Беляев. — М : Радио и связь, 1985. — 608 с.

28. Бенсусан А. Методы декомпозиции, децентрализа-ции, координации и их приложения [Текст] / А. Бенсусан, Ж.Л. Лионс, Р. Темам // Методы вычислительной математики — Новосибирск : Наука, 1975.

29. Берндт Т. Сортировочные горки на железных дорогах мира [Текст] / Т. Берндт, С. В. Власенко // Автоматика, связь, информатика. 2007. № 6. — С. 45-48.

30. Бобровский В.И. Оптимизация режимов торможения отцепов на сортировочных горках : Монография / В.И. Бобровский и др. — Дн-вск : Изд-во Маковецкий, 2010. — 260 с.

31. Бондарев В.М. Основы программирования [Текст] / В.М. Бондарев, В.И. Рублинецкий, Е.Г. Качко. — Харьков : Фолио, 1998. — 256 с.

32. Босов А.А. Структурная сложность систем [Текст] / А. А. Босов , В. М. Ильман // Вісник ДНУЗТ ім. ак. В. Лазаряна. 2012. № 40. — С. 173-179.

33. Бочаров А.П. Разработка и внедрение корпоративной информационной системы: основные положения и проблемы [Текст] / А. П. Бочаров, П. П. Науменко, Ф. А. Карбинский, В. А. Шиш // Інформаційно-керуючі системи на залізничному транспорті. 2012. № 5. — С. 3-8.

34. Бочаров О.П. Динамічна модель сортувальної станції та її роль в подальшій оптимізації процесу перевезень [Текст] / О. П. Бочаров, Г. О. Міхальов, В. П. Мороз, В. О. Шиш // Інформаційно-керуючі системи на залізничному транспорті. 2011. № 5. — С. 74-76.

35. Буянов В.А. Автоматизированные информационные системы на железнодорожном транспорте [Текст] / В.А. Буянов, Г.С. Ратин. — М : Транспорт, 1984. — 239 с.

36. Вагин В.Н. Достоверный и правдоподобный вывод в интеллектуальных системах [Текст] / В. Н. Вагин, Е. Ю. Головина, А.А. Загорянская, М. В. Фомина —М. : ФИЗМАТЛИТ, 2004. — 704 с.

37. Вальков В.М. Микроэлектронные управляющие вычислительные комплексы. Системное проектирование и конструирование [Текст] / В.М. Вальков. — Л : Машиностроение. Ленингр. отд-ние, 1990. — 224 с.

38. Великодный В.В. Задачи по эксплуатации вагонных парков на основе ав-томатизированной системы управления грузовыми перевозками Укрзализныци [Текст] / В. В. Великодный, В. Б. Землянов, В. В. Скалозуб, И. В. Жуковицкий, С. Ю. Цейтлин // Інформаційно-керуючі системи на залізничному транспорті. 2005. № 3. — С. 31-35.

39. Гельфанд И.М. О некоторых способах управления сложными системами [Текст] / И. М. Гельфанд, М.Л. Цетлин // Успехи математических наук. 1962. Т. 17. № 1 (103). — С. 3-25.

40. Гильман А.С. Учет характеристик программного обеспечения при анализе надежности УВК АСУТП на этапе проектирования [Текст] / А.С. Гильман // Приборы и системы управления. 1982. № 3. — С. 8-11.

41. Де Гроот М. Оптимальные статистические решения [Текст] / М. Де Гроот. — М. : Мир, 1974.

42. Древс Ю.Г. Системы реального времени: технические и программные средства [Текст] / Ю.Г. Древс. — М : МИФИ, 2010. — 320 с.

43. Душкин Д.Н. Сетецентрические технологии: эволюция, текущее положение и области дальнейших исследований [Текст] / Д. Н. Душкин, М. П. Фархадов // Автоматизация и современные технологии. - М.: Машиностроение. 2012. № 1. — С. 21-29.

44. Жуковицький І.В. Розвиток теорії та удосконалення систем автоматичного управління швидкістю скочування відчепів на сортувальних гірках [Текст] : автореф. дис. .... д.т.н. 05.22.08 / Жуковицький Ігор Володимирович. — Харків : Харківська державна академія залізничного транспорту, 1999. — 34 с.

45. Забегалин Е.В. Архитектура информационных систем в теории и практике. IBS. Департамент управленческого консалтинга [Электронный ресурс] / Е. В. Забегалин // Режим доступа : http://www.evz.name/evzms-2.pdf. 2012.

46. Зиглер К. Методы проектирования программных систем, Пер. с англ. [Текст] / К. Зиглер. — М : Мир, 1985. — 328 с.

47. Иванченко В.И. Новый подход к управлению процессом роспуска составов на сортировочной горке [Текст] / В. И. Иванченко, Н. Н. Лябах, А. А. Сепетый // Труды РИИЖТа. - Ростов-на-Дону. 1984. — С. 34-41.

48. Иванченко В.М. Микропроцессорные информационно-управляющие системы автоматизации сортировочных процессов. Уч. пособие [Текст] / В. М. Иванченко. — Ростов-на-Дону, 1984. — 96 с.

49. Иванченко В.Н. Теория построения и реализация информационно-управляющих микропроцессорных систем на сортировочных станциях [Текст] : автореф. дис. .... д.т.н. 05.22.08 / Иванченко Владимир Николаевич. — Ленинград : ЛИИЖТ имени академика В. Н. Образцова, 1988. — 48 с.

50. Иванюто И.Д. АСУ сортировочными станциями (на примере АСУ СС НПО «Агат») [Текст] / И. Д. Иванюто, С. С. Галуза, А. А. Ерофеев, Н. Н. Казаков —Гомель : Белорусский государственный университет транспорта, 2003. — 159 с.

51. Ишков П.В. Виртуализация: основа построения эффективных ИТ [Текст] / П. В. Ишков // Автоматика, связь, информатика. 2012. № 2. — С. 11-13.

52. Івченко Ю.М. Інтеграція мережевого обладнання АСК ВП УЗ та АСК ПП УЗ, підключення його до ЄМПД [Текст] / Ю. М. Івченко, В. Г. Івченко, О. М. Гондар // Вісник ДНУЗТ ім. ак. В. Лазаряна // 2009. № 29. — С. 143-146.

53. Казиев Г.Д. Задачи технического перевооружения сортировочных станций [Текст] / Г. Д. Казиев, А. Г. Савицкий // Автоматика, связь, информатика. 2007. № 4. — С. 17-22.

54. Каменнова М.С. Системный подход к проектированию сложных систем [Текст] / М.С. Каменнова // Журнал д-ра Добба. 1993. № 1. — С. 9-14.

55. Кинаш С.А. Унификация архитектурных решений для АСУ РЖД [Текст] / С. А. Кинаш // Автоматика, связь, информатика. 2007. № 7. — С. 20-21.

56. Кириллов В.П. SSADM - передовая технология разработки автоматизированных систем [Текст] / В.П. Кириллов // Компьютеры + Программы. 1994. № 2(10). — С. 8-16.

57. Клир Д. Системология. Автоматизация решения системных задач : Пер. с англ. / Дж. Клир. — М : Радио и связь, 1990. — 544 с.

58. Козлов Б.А. Справочник по расчету надежности аппаратуры радиоэлектроники и автоматики [Текст] / Б.А. Козлов, И.А. Ушаков. — М : Сов.радио, 1975. — 472 с.

59. Косолапов А.А. Автоматизация инженерного проектирования информационно-управляющих вычислительных систем и сетей: методика, модели, методы, программная среда [Текст] / А.А. Косолапов. — Винница : ВПИ, ИК им. ак. В.М.Глушкова, АН Украины, 1993. — С. 114.

60. Косолапов А.А. Автоматизация системного проектирования информационно-управляющих вычислительных систем и сетей АСУТП [Текст] / А.А. Косолапов // Комп’ютерне моделювання. Дніпродзержинськ : Мін. освіти, ДДТУ 1998. — С. 96-97.

61. Косолапов А.А. Автоматизированная система управления маршрутами движения для сортировочных горок на базе микропроцессорной ЭВМ типа СМ1800 [Текст] / Г. А. Красовский, Б. А. Игнатов, Л. В. Сафрис, Б. М. Филимонов, А. А. Косолапов // Автоматика, телемеханика и связь. 1984. № 7. — С. 7-11.

62. Косолапов А.А. Анализ состояния и перспектив развития средств технической диагностики управляющих вычислительных комплексов в АСУТП [Текст] / А.А. Косолапов, В.Г. Савченко // Автоматизированные информационные системы на сортировочных станциях железнодорожного транспорта. - Межвуз. сб. научн. трудов. - Днепропетровск: изд. ДИИТа. 1988. № 263/16. — С. 5-12.

63. Косолапов А.А. Внедрение информационных технологий для обеспечения качества подготовки специалистов. ЦИТ: 212-725 [Текст] / Б. Е. Боднарь, Е. Б. Боднарь, А. А. Косолапов // Сб. научных трудов SWorld "Перспективные инновации в науке, образовании, производстве и транспорте '2012". Технические науки. - Одесса. 2012. Т. 5. № 2. — С. 31-36.

64. Косолапов А.А. Выбор технических средств и разработка вариантов технической структуры СКАТ-СС [Текст] / Е. М. Шафит, И. В. Жуковицкий, А. А. Косолапов, Т. В. Туник // Інформ.-керуючі системи на залізн. трансп. 2000. № 2. — С. 14-19.

65. Косолапов А.А. Дослідження умов доцільності децентралізації функцій керування в системах гіркової автоматики [Текст] / А.А. Косолапов. ЦИТ: 213-903 // Сборник научных трудов SWorld. 2013. Т. 2. № 2. — С. 37-45.

66. Косолапов А.А. Информационно-методическая база управления учебным процессом в техническом университете [Текст] / А. Н. Пшинько, А. С. Распопов, А. А. Косолапов // Информационная инфраструктура высших учебных заведений. - Сборник научных трудов. 1999. Т. 2. — С. 85-88.

67. Косолапов А.А. Исследование механизмов защиты систем в условиях сбоев и деградации с использованием иерархических сетей Петри [Текст] / А. А. Косолапов, М. А. Демчук // Вісник Технологічного університету Поділля. Технічні науки. Науковий журнал, Хмельницький. 2003. Т. 2. № 3. — С. 64-67.

68. Косолапов А.А. К вопросу структурного проектирования автоматизированных систем [Текст] / А. А. Косолапов, М. А. Косолапова // Математичне моделювання. Науковий журнал. Дніпродзержинськ: - Мін. освіти, ДДТУ. 2000. № № 2(5). — С. 124-128.

69. Косолапов А.А. Ключевая роль транспорта в современном мире : монография [Текст] / [авт. кол. : Косолапов А. А., Блохин А. Л., Боряк К. Ф. и др.]. — Одесса : КУПРИЕНКО СВ, 2013. — 163 с. - ISBN 978-966-2769-16-6.

70. Косолапов А.А. Концептуальні моделі гіркових процесів і систем. ЦИТ: 412-0950. [Текст] / А. А. Косолапов // Сборник научных трудов SWorld. Материалы международной научно-практической конференции «Современные проблемы и пути их решения в нау-ке, транспорте, производстве и образовании'2012». - Технические науки. Информатика, вычислительная техника и автоматизация. - Одесса: КУПРИЕНКО. 2012. Т. 14. № 4. — С. 71-81.

71. Косолапов А.А. Концептуальні моделі сортувальних станцій. ЦИТ: 412-0949. [Текст] / А. А. Косолапов // Сборник научных трудов SWorld. Материалы международной научно-практической конференции «Современные проблемы и пути их решения в науке, транспорте, производстве и образовании'2012». - Транспорт. Техническая эксплуатация и ремонт средств транспорта. - Одесса: КУПРИЕНКО. 2012. Т. 2. № 4. — С. 65-69.

72. Косолапов А.А. Концепція побудови, розробки і впровадження автоматизованої інформаційної системи управління університетом на базі мережевих інформаційних інтернет-технологій [Текст] / О. М. Пшінько, В. Я. Нечай, Є.М.Шафіт, А.А. Косолапов // Інформ.-керуючі системи на залізн. трансп. 1998. № 4. — С. 70-71.

73. Косолапов А.А. Логіко-лінгвістична система управління сповільнювачем прицільної гальмівної позиції на сортувальній гірці. ЦИТ: 212-750 [Текст] / А. А. Косолапов // Сб. научных трудов SWorld. "Перспективные инновации в науке, образовании, производстве и транспорте '2012 ". - Транспорт. Физика и математика. - Одесса. 2012. Т. 2. № 2. — С. 58-61.

74. Косолапов А.А. Машинная методика эскизной компоновки и расчета характеристик децентрализованных УВК систем горочной автоматики на МКЭВМ [Текст] / А.А. Косолапов, О.В. Рожков // Межвуз. сб. научн. трудов. - Днепропетровск: изд ДИИТа. 1984. № 236/13. — С. 24-29.

75. Косолапов А.А. Методика анализа надёжности нечётких систем с использованием теории размытых множеств. ЦИТ: 113-1008. [Текст] / А. А. Косолапов // Сборник научных трудов SWorld. Материалы международной научно-практической конференции «Современные направления теоретических и прикладных исследований ‘2013». - Технические науки. Информатика, вычислительная техника и автоматизация. - Одесса: КУПРИЕНКО. 2013. Т. 10. № 1. — С. 24-35.

76. Косолапов А.А. Методика выбора технических структур управляющих систем сортировочных горок по критерию эффективного использования вычислительных ресурсов [Текст] / А.А. Косолапов // (в печати). 2013.

77. Косолапов А.А. Методика разработки, структура и характеристика микропроцессорной информационно-управляющей вычислительной системы для ГАЦ (АСУ МД) : сборник научных трудов [Текст] / Е. М. Шафит, Л. В. Сафрис, А. А. Косолапов // Автоматизация управления маршрутами на сортировочных горках : Сб. науч. тр. ВНИИЖТ. М. 1984. — С. 70-78.

78. Косолапов А.А. Методология автоматизированного системного анализа и проектирования как основа создания информационно-управляющих вычислительных систем и сетей / А. А. Косолапов // Информационные технологии на железнодорожном транспорте “ИНФОТРАНС-96”. — СПб. : ПГУПС, 1996. — С. 332-341.

79. Косолапов А.А. Методологія проектування інтегрованих комп'ютерних систем управління організаційно-технологічними процесами на залізничному транспорті [Текст] / А. А. Косолапов // Інформ.-керуючі системи на залізн. трансп. . 1998. № 4. — С. 69-70.

80. Косолапов А.А. Многоуровневая структурная оптимизация в составе инженерной методики проектирования корпоративных информационных систем [Текст] / А. А. Косолапов // Математичне моделювання. Науковий журнал. Дніпродзержинськ: - Мін. освіти, ДДТУ. 2000. № 1(4). — С. 57-60.

81. Косолапов А.А. Многоуровневая структурная оптимизация в составе инженерной методики проектирования корпоративных информационных систем [Текст] / А. А. Косолапов // Автоматика-2000. Міжнародна конференція з автоматичного управління, Львів, 11-15 вересня 2000: Праці в 7-ми томах.- Львів: Державний НДІ інформаційної інфраструктури. 2000. Т. 6. — С. 274-278.

82. Косолапов А.А. Моделі дискретних систем реального масштабу часу керування сортувальними гірками. ЦИТ: 312-793 [Текст] / А. А. Косолапов // Сб. научных трудов SWorld. «Научные исследования и их практическое применение. Современное состояние и пути развития `2012». - Транспорт. Физика и математика. - Одесса. 2012. Т. 2. — С. 46-58.

83. Косолапов А.А. Некоторые характеристики микропроцессоров и микро-ЭВМ и технологических алгоритмов в системах управления роспуском составов на горке [Текст] / Е.М. Шафит, Л.В. Сафрис, А.А. Косолапов // Труды. Межвузовский тематический сборник. Ростов-на-Дону. РИИЖТ. 1982. № 168. — С. 20-23.

84. Косолапов А.А., Пшінько Ю.О. Онтологічні моделі в задачах автоматизації сортувальних станцій [Текст] / А.А. Косолапов, Ю.О. Пшінько // Сборник научных трудов SWorld. Материалы международной научно-практической конференции «Научные исследования и их практическое применение. Современное состояние и пути развития ‘2013». – Выпуск 4. Том 12. Транспорт. Техническая эксплуатация и ремонт средств транспорта. – Одесса: КУПРИЕНКО, 2013. – ЦИТ: 410-0947. – С. 65-69.

85. Косолапов А.А. Оптимизация распределения информационных потоков при проектировании автоматизированных систем / А. А. Косолапов // Оптимизация производственных процессов. - Сб. научн. трудов. Севастополь: изд. СевГТУ. 2000. № 3. — С. 143-146.

86. Косолапов А.А. Организация антивирусной защиты и самодиагностики в распределённых информационных системах [Текст] / А. А. Косолапов, Д. В. Лоскутов, Н. А. Фастов // Інформаційні технології в наукових дослідженнях і навчальному процесі : матеріали V Міжнар. наук.-практ. конф. (17-19 листоп. 2010 р.). - Луганськ. 2010. — С. 176-181.

87. Косолапов А.А. Организация подготовки специалистов в области современных сетевых информационных технологий для железных дорог Украины [Текст] : материалы временных коллективов / В. А. Домашний, В. С. Юденко, Е. М. Шафит, А. А. Косолапов. — СПб, 1996. — С. 259-263.

88. Косолапов А.А. Организация управления учебным процессом на основе информационной системы университета [Текст] / А. Н. Пшинько, А. С. Распопов, А. А. Косолапов // 3 Міжнародна науково-методична конференція "Освіта та Віртуальність". Наукові праці. — Харків : ХТУРЕ, 1999. — С. 68-70.

89. Косолапов А.А. Основные направления и этапы развития систем управления сортировочным процессом на станциях [Текст] / Е.М. Шафит, Л.В. Сафрис, А.А. Косолапов // Автоматизированные системы управления технологическими процессами на сортировочных станциях магистрального и промышленного транспорта. - Межвуз. сб. научн. трудов. - Днепропетровск: изд. ДИИТа. 1985. № 244/14. — С. 3-16.

90. Косолапов А.А. Оценка надежности автоматизированной системы управления маршрутами движения на горке (АСУМД) [Текст] / Л.А. Вашурин, В.Н. Рыбцов, С.Г. Чижов, А.А. Косолапов // Автоматизированные системы управления технологическими процессами на сортировочных станциях магистрального и промышленного транспорта. - Межвуз. сб. научн. трудов. - Днепропетровск: изд. ДИИТа. 1985. № 244/14. — С. 52-62.

91. Косолапов А.А. Подсистема диагностики и обслуживания многомикромашинных информационно-управляющих вычислительных систем автоматизации сортировочных станций [Текст] / А.А. Косолапов, Л.Г. Богдан, А.А. Гарибян // Микропроцессорные системы и устройства управления ответственными технологическими процессами на транспорте. - М.: ЦП ВНТО ж.д. и трансп. строителей, МИИТ. 1989. — С. 45-47.

92. Косолапов А.А. Подсистема контроля, диагностики и обслуживания комплекса технических средств ИПС СС [Текст] / Е.М. Шафит, А.А. Косолапов, Л.Г. Богдан, Н.И. Бубело // Межвуз. сб. научн. трудов. - Днепропетровск: изд. ДИИТа. 1990. № 279/17. — С. 5-10.

93. Косолапов А.А. Подход к оценке влияния диагностики вычислительной системы на показатели ее надежности [Текст] / А.А. Косолапов, В.Г. Савченко // Автоматизированные информационные системы на сортировочных станциях железнодорожного транспорта. - Межвуз. сб. научн. трудов. - Днепропетровск: изд. ДИИТа. 1988. № 263/16. — С. 13-22.

94. Косолапов А.А. Пошук оптимальної структури інформаційної системи. Генетичний підхід. ЦИТ: 113-1007. [Текст] / А. А. Косолапов, Ю. О. Гриненко // Сборник научных трудов SWorld. Материалы международной научно-практической конференции «Современные направления теоретических и прикладных исследований ‘2013». - Транспорт. Транспортные и логистические системы. - Одесса: КУПРИЕНКО. 2013. Т. 1. № 1. — С. 35-44.

95. Косолапов А.А. Применение микропроцессорных средств в автоматизированных системах управления сортировочным процессом на горке [Текст] / Е.М. Шафит, Л.В. Сафрис, А.А. Косолапов // Автоматизированные системы управления технологическими процессами на железнодорожных станциях : Межвуз. сб .науч. тр. - Днепропетровск : ДИИТ. 1982. № 224/11. — С. 3-12.

96. Косолапов А.А. Принципы декомпозиции и интеграции автоматизированных систем управления сортировочными станциями [Текст] / Е.М. Шафит, А.А. Косолапов // Автоматизированные информационно-управляющие вычислительные системы на сортировочных станциях железнодорожного транспорта. - Межвуз. сб. научн. трудов. - Днепропетровск: изд. ДИИТа. 1993. № 289/18. — С. 5-17.

97. Косолапов А.А. Принципы нечёткой маршрутизации. ЦИТ: 312-792 [Текст] / А. А. Косолапов // Сб. научных трудов SWorld. «Научные исследования и их практическое применение. Современное состояние и пути развития `2012». - Технические науки. - Одесса. 2012. Т. 6. — С. 18-22.

98. Косолапов А.А. Принципы описания общего алгоритма функционирования децентрализованных систем управления технологическими процессами [Текст] / Е.М. Шафит, Л.В. Сафрис, А.А. Косолапов, Ю.А. Косорига // Автоматизированные системы управления технологическими процессами на железнодорожных станциях. Межвуз. сб. науч. тр. - Днепропетровск : ДИИТ. 1983. — С. 28-36.

99. Косолапов А.А. Принципы построения интегрированной автоматизированной системы управления технологическими процессами на сортировочной станции / Е. М. Шафит, И. В. Жуковицкий, А. А, Косолапов // Информационно-управляющие системы на железнодорожном транспорте. 1996. № 1,2. — С. 36-41.

100. Косолапов А.А. Принципы проектирования и структура корпоративной информационной системы университета / А. А. Косолапов // 3 Міжнародна науково-методична конференція "Освіта та Віртуальність". Наукові праці - 104 с. // ХТУРЕ — Харків, 1999. — С. 64-67.

101. Косолапов А.А. Проектування інтелектуальних транспортних систем і мереж засобами аналітичних моделей масового обслуговування [Текст]: Монографія / А. А. Косолапов. — Дніпропетровськ : Вид-во Дніпропетр. нац. ун-ту залізн. трансп. ім. акад. В. Лазаряна, 2013. — 190 с.

102. Косолапов А.А. Развитие технической структуры автоматизированных систем управления технологическими процессами на сортировочных станциях. Система СКАТ-СС / Е. М. Шафит, И. В. Жуковицкий, А. А. Косолапов // Proceedings JУЖЕЛ 7th International Conference of Railway Experts. Yugoslavia, Vrnjacka Banja, October 04-06. 2000. — С. 145-146.

103. Косолапов А.А. Развитие технической структуры управляющих вычислительных комплексов АСУ МД на базе микроЭВМ СМ1810 : сборник [Текст] / Е.М. Шафит, А.А. Косолапов // Развитие АСУ маршрутами движения на сортировочных горках. - М. 1987. — С. 65-67.

104. Косолапов А.А. Разработка и опыт применения средств автоматизации проектирования локальных вычислительных систем АСУ [Текст] / Е.М. Шафит, А.А. Косолапов // Практическое применение современных технологий программирования, пакетов прикладных программ в вычислительных системах и сетях ЭВМ. Инф. сборник “Приборы, средства автоматизации и системы управления“. Часть 2. - Серия “Научно-техническая пропаганда“. - М.: Информприбор. 1988. № 8. — С. 5-7.

105. Косолапов А.А. Разработка и применение методов математического моделирования при анализе и проектировании микропроцессорных АСУ технологическими процессами роспуска составов на горках [Текст] : автореф. дис. ... к.т.н. 05.13.07 / Косолапов Анатолий Аркадьевич —Москва : Всесоюзный ордена Трудового Красного Знамени научно-исследовательский институт железнодорожного транспорта (ЦНИИ МПС), 1984. — 24 с.

106. Косолапов А.А. Разработка подсистемы диагностики микрокомпьютерных локальных вычислительных сетей АСУ [Текст] / А. А. Косолапов, Л. Г. Богдан, В. Н. Саква, А. П. Тихонов, Н. И. Бубело // Практическое применение современных технологий программирования, пакетов прикладных программ в вычислительных системах и сетях ЭВМ. - Инф. сборник “Приборы, средства автоматизации и системы управления“. Часть 2. - Серия “Научно-техническая пропаганда“. - М.: Информприбор. 1988. № 8. — С. 30-31.

107. Косолапов А.А. Разработка УВК на базе микро-ЭВМ для совершенствования систем горочной автоматики и сопряжение их с АСУСС [Текст] / А.Н. Вахнин, Т.Н. Зиброва, А.А. Косолапов // Автоматизированные системы управления технологическими процессами на железнодорожных станциях: Межвуз. сб. науч. тр. - Днепропетровск : ДИИТ. 1983. № 224/11. — С. 3-12.

108. Косолапов А.А. Разработка унифицированной системы автоматизированного управления процессом расформирования поездов на сортировочной станции [Текст]: материалы временных коллективов / Е. М. Шафит, И. В. Жуковицкий, А. А. Косолапов. — Санкт-Петербург,, 1996. — С. 81-93.

109. Косолапов А.А. Резервы архитектуры автоматизированной системы управления грузовыми перевозками Украинских железных дорог [Текст] / А. А. Косолапов, И. В. Жуковицкий // Залізничний транспорт України. - Київ, 2013, № 1. — С. 10-13.

110. Косолапов А.А. Системи штучного інтелекту [Текст]: методичні вказівки до практичних і лабораторних робіт із розділу «Нечітке виведення в інтелектуальних системах проектування та управління» / уклад. А. А. Косолапов. — Дніпропетровськ : Вид-во Дніпропетр. нац. ун-ту залізн. трансп. ім. акад. В. Лазаряна, 2012. — 47 с.

111. Косолапов А.А. Системные принципы построения технической структуры автоматизированной информационной системы университета [Текст] / А. А. Косолапов // Информационная инфраструктура высших учебных заведений. - Сборник научных трудов. -Санкт-Петербург. - 98 с. 1999. Т. 2. — С. 48-50.

112. Косолапов А.А. Системотехнический подход к построению структуры интегрированных автоматизированных систем управления сортировочными станциями [Текст] / А.А. Косолапов, Е.М. Шафит // Микропроцессорные системы и устройства управления ответственными технологическими процессами на транспорте. - М.: ЦП ВНТО ж.д. и трансп. строителей, МИИТ 1989. — С. 42-44.

113. Косолапов А.А. Таблично функциональное моделирование в задачах проектирования управляющих вычислительных систем автоматизации сортировочных станций [Текст] / А.А. Косолапов // Некоторые вопросы математического моделирования в инженерных задачах. - Сб. научн. трудов. - Днепропетровск: изд. ДИИТа. 1994. № 297/3. — С. 66-74.

114. Косолапов А.А. Тенденції розвитку архітектури автоматизованих систем керування [Текст] / А. А. Косолапов, І. В. Жуковицький // Системные технологии. Региональный межвузовский сборник научных работ. 2013. № 3 (86). — С. 62-71.

115. Косолапов А.А. Формирование рациональной структуры телекоммуникационной среды корпорации на основе эволюционно-генетического подхода [Текст] / А. А. Косолапов, А. Б. Мудрик // Информационные технологии на железнодо-рожном транспорте. Доклады десятой международной научно-практической конференции „Инфотранс-2005". - СПб. 2005. — С. 135-140.

116. Косолапов А.А., Рыбцов В.Н. Оценка эффективности функционирования отдельных вариантов структуры микропроцессорных АСУ роспуском составов на горке с учётом надёжности вычислительного комплекса [Текст] / А.А. Косолапов, В.Н. Рыбцов // В. сб.: Автоматизированные системы управления технологическими процессами на железнодорожных станциях. - Днепропетровск: ДИИТ. 1982. — С. 81-91.

117. Крицкий С.П. Трансляция языков программирования: синтаксис, семантика, перевод. [Електронний ресурс] // Режим доступу : http://public.uic.rsu.ru/~skritski/scourses/Transl/index.html 2013.

118. Кузнецов С. Переносимость и интероперабельность информационных систем и международные стандарты [Текст] / С. Кузнецов // Computer World. 1996. № 4. — С. 5-6.

119. Леоненков А.В. Нечёткое моделирование в среде MatLab и fuzzyTech [Текст] / А. В. Леоненков. — СПб : БХВ-Петербург, 2003. — 736 с.

120. Лоскутов А.Ю. Основы теории сложных систем [Текст] / А. Ю. Лоскутов, А. С. Михайлов —М.-Ижевск : НИЦ «Регулярная и стохастическая динамика», 2007. — 620 с.

121. Майоров С.А. Основы теории вычислительных систем [Текст] / С.А. Майоров, Г.И. Новиков, Алив Т.И. и др. Учеб. пособие для ВУЗов. — М : Высшая школа, 1978. — 408 с.

122. Меньшиков Н.Я. Надёжность железнодорожных систем автоматики [Текст] / Н.Я. Меньшиков, А.И. Королёв, Р.Ш. Ягудин —М: Транспорт, 1976. — 215 с.

123. Месарович М. Теория иерархических многоуровневых систем : Пер. с англ. [Текст] / М. Месарович, Д. Мако, И. Такахара. — М. : "Мир", 1973. — 344 с.

124. Мішечкін В. Історія автоматизації перевізних процесів / В. Мішечкін // 2013. - [Электронный ресурс]. - Режим доступа : URL: http://www.youtube.com/watch?v=FzBG7-Pkx4M&feature=relmfu/. 2013.

125. Нейгел К. C# 2005 и платформа .NET 3.0 для профессионалов [Текст] / Кристиан Нейгел, Билл Ивьен, Джей Глинн, Морган Скиннер, Карли Уотсон. — М : Диалектика, 2008. — 1376 с.

126. Нечипоренко В.И. Структурный анализ систем (эффективность и надёжность) [Текст] / В. И. Нечипоренко. — М. : Сов. радио, 1977. — 216 с.

127. Плотникова А. АСК ВП УЗ-Е – 20 дней спустя / А. Плотникова // "Магистраль", 01.08.2012. - [Электронный ресурс]. - Режим доступа : URL: www.magistral-uz.com.ua/. 2012.

128. Половко А.М. Основы теории надёжности [Текс] / А. М. Половко, С. В. Гуров. — Спб. : БХВ-Петербург, 2008. — 704 с.

129. Растригин Л.А. Адаптация сложных систем [Текст] / Л.А. Растригин. — Рига : Зинатне, 1981.

130. Савицкий А.Г. Комплексная система автоматизированного управления сортировочной станцией [Текст] / А. Г. Савицкий // Евразия Вести. 2004. № XI.

131. Савицкий А.Г. Комплексная система автоматизированного управления сортировочным процессом [Текст] : автореф. дис. .... к.т.н. 05.22.08 / Савицкий Александр Григорьевич. — Москва : Московский государственный университет путей сообщения (МИИТ), 2005. — 24 с.

132. Самков А.Н. К вопросу о выборе УВК для АСУРСГ [Текст] / А.Н. Самков, Е.М. Шафит // Труды ДИИТа. 1980. № 211/9. — С. 11-17.

133. Сафрис Л.В. Концептуальная модель автоматизированной системы управления роспуском составов на горке [Текст] / Л.В. Сафрис // Автоматизированные системы управления технологическими процессами на железнодорожных станциях. Межвуз. сб. научн. трудов. - Днепропетровск: изд. ДИИТа. 1981. № 218/10. — С. 8-14.

134. Сафрис Л.В. Об одном критерии автоматического выбора оптимальной скорости роспуска [Текст] / Л.В. Сафрис // Труды ДИИТа Т. 121/3. — Днепропетровск : ДИИТ, 1971.

135. Сафрис Л.В. Стратегия целевого торможения при автоматизации сортировочной горки [Текст] / Л.В. Сафрис // Труды ДИИТа Т. 162/6. — Днепропетровск : ДИИТ, 1974.

136. Сергиенко Н.И. Проектный подход при внедрении информационных технологий и систем для железнодорожного транспорта Украины [Текст] / Н.И. Сергиенко, О.Л. Зиненко // Современные информационные технологии на транспорте, в промышленности и образовании. Материалы Междунар. научно-практ. конф., г. Днепропетровск, 18-19 апреля 2013 г. 2013. — С. 87.

137. Сидорова Е.Н. Автоматизированные системы управления в эксплуатационной работе [Текст] / Е. Н. Сидорова. — М. : Маршрут, 2005. — 560 с.

138. Стукалов Е.А. Определение параметров потока инициативных сигналов, поступающих в УВК АСУ расформированием составов на сортировочной горке [Текст] / Е.А. Стукалов // Труды МИИТа. 1983. № 719. — С. 93-99.

139. Трифонов П.В. Информатика. Построение и анализ алгоритмов : Учебное пособие для ВУЗов [Текст] / П.В. Трифонов. — Санкт-Петербург : Питер, 2007. — 95 с.

140. Трутнев Д.Р. Архитектуры информационных систем. Основы проектирования : Учебное пособие [Текст] / Д. Р. Трутнев. — СПб : НИУ ИТМО, 2012. — 66 с.

141. Туманова Т. Единый Центр обработки данных заработал в Укрзализныце [Текст] / Т. Туманова // УНН. - 2012. - [Электронный ресурс]. - Режим доступа : URL: http://www.unn.com.ua/. 2012.

142. Федюшин Ю.М. Информатизация железнодорожного транспорта Украины / Ю. М. Федюшин // Информационно-управляющие системы на железнодо-рожном транспорте. 1997. № 4. — С. 3-5.

143. Фонарев Н.М. Автоматизация процесса расформирования составов на сортировочных горках [Текст] / Н.М. Фонарев. — М : Транспорт, 1971. — 271 с.

144. Фритч В. Применение микропроцессоров в системах управления : Пер. с нем. [Текст] / В. Фритч. — М. : Мир, 1984. — 464 с.

145. Хакен Г. Синергетика. Иерархии неустойчивостей в самоорганизующихся системах и устройствах. [Текст] / Г. Хакен —М. : Мир, 1985.

146. Хетагуров Я.А. Проектирование информационно-вычислительных комплексов [Текст] // Я.А. Хетагуров, Ю.Г. Древс —М. : Высш. шк., 1987. — 280 с.

147. Цейтлин С.Ю. Типовые проектные решения для создания АСУ ВП УЗ-Е [Текст] / С. Ю. Цейтлин, В. К. Башлаев // Тезисы Международной научно-практической конференции "Современные информационные технологии на транспорте, в промышленности и образовании" (15.05.2008-16.05.2008), Днепропетровск. 2008. — С. 31-32.

148. Цилькер Б.Я. Организация ЭВМ и систем : Учебник для ВУЗов [Текст] / Б.Я. Цилькер, С.А. Орлов. — Санкт-Петербург : Питер, 2006. — 654 с.

149. Чистяков Л.С. Модифицированная HIPO-технология разработки больших программных комплексов [Текст] / Л.С. Чистяков // Прикладная информатика. 2006. № 6. — С. 64.

150. Шастова Г.А., Коёкин А. И. Выбор и оптимизация структуры информационных систем [Текст] / Г. А. Шастова, А. И. Коёкин —М. : Энергия, 1972. — 256 с.

151. Шафит Е.М. Исследование влияния надежности на показатели качества функционирования сложных систем [Текс] / Е.М. Шафит, А.Н. Самков, В.Н. Рыбцов // Автоматизированные системы управления технологическими процессами на железнодорожных станциях: Межвуз. сб. научн. тр. - Днепропетровск : ДИИТ. 1984. — С. 3-10.

152. Шафит Е.М. Экономическая эффективность и надежность систем автоматического управления сортировочным процессом с управляющей ЦВМ [Текст] / Е.М. Шафит , А.Н. Самков // Труды ДИИТа. 1971. № 129/3. — С. 3-20.

153. Шлеер С. Объектно-ориентировочный анализ: моделирование мира в состояниях : Пер. с англ. [Текст] / С. Шлеер, С. Меллор. — Киев : Диалектика, 1993. — 240 с.

154. Юнг М. Сименс – партнёр для автоматизации сортировочных станций [Текст] / М. Юнг, О. В. Подсосонная // Автоматика, связь, информатика. 2009. № 1. — С. 51-52.

155. Baun C. Cloud Computing. Web-Based Dynamic IT Services / C. Baun. — Berlin : Springer-Verlag, 2011. — 108 p.

156. Cheng C.H., and Mon, D. L. . Fuzzy system reliability analysis by interval of confidence // Fuzzy Sets and Systems. 1993. Т. 56. № 1. — С. 29-35.

157. Dave P. System Design Strategies // An ESRI White Paper. February. 2003.

158. Automation Device and Computing 2012-2013 [Електронний ресурс] / Режим доступу : http://support.advantech.com.tw/ OnlineResources/ SubIndex.aspx?bu=99B2E2BE-A7E3-4C58-9E35-B4F48A952AC7&type =e Catalog. : Advantech, 2013.

159. Definition issue des travaux du groupe de travail Interop de I'AFUL [Электронный ресурс] // Режим доступа: http:// http://aful.org/gdt/interop. 2013.

160. The MSR 32 microcomputer system. Greater efficiency and safety for cargo transport [Электронный ресурс] / Siemens AG // Режим доступа : http://www.siemens.com/mobility. 2008.

161. Produre, method and organization for the development of international Computer system within the UIC. PROMETEO / UIC Code 902. — Paris : Int. Union of Railways, 1993. — 138 p.

162. The Protégé Ontology Editor [Электрон. ресурс] / Режим доступа: http://protege.stanford.edu/. — Manchester, 2013.

163. Real Time Systems Modeling, Design and Applications. AMAST Series in Computing // World scientific. 2007. Т. 8. — С. 37-62.

164. A Survey of System Development Process Models CTG.MFA – 003 Models for Action Project : Developing Practical Approaches to Electronic Records Management and Preservation. — Center for Technology in Government University at Albany : SUNY, 1998. — 13 p.

165. SYSTEM DEVELOPMENT METHODOLOGY (SDM) September 6, Version 2, 02.18.2005. — NH Office of Information Technology : Agency Software Division (ASD) Effective, 2006. — 35 p.

166. Adamek J., Trnkova V. Automata and algebras in categories. : Kluwer Academic Publisher. 1989.

167. Arbib M.A. Arrows, structures and functors : The categorical imperative / M. A. Arbib & E. G. Manes. : Academic Press, 1975. — 320 p.

168. Arbib M.A. Theories of Abstract Automata // Series in Automatic Computation — Englewood Cliffs, NJ : Prentice Hall, 1969.

169. Atanassov K. Intuitionistic fuzzy sets / K. Atanassov // Fuzzy Sets and Systems. 1986. № 20. — С. 87 - 96.

170. Baeten J.C.M. Real time process algebra / J. C. M. Baeten & J. A. Bergstra // Formal Aspects of Computing. 1993. Т. 3. — С. 142-188.

171. Bertoli M. The JMT simulator for performance evaluationof non-product-form queueing networks / M. Bertoli, G. Casale, G. Serazzi // Proc. of the 40th Annual Simulationymposium (ANSS). 2007. — С. 3-10.

172. Brandin B.A. Supervisory control of timed discrete-event systems / B. A. Brandin & M. W. Wonham // IEEE Transactions on Automatic Control. 1994. Т. 39(2). — С. 329-343.

173. Buchi R.J. Finite automata, their algebras and grammars / R. J. Buchi. : Springer-Verlag, 1989. — 434 p.

174. Cai K.Y. Posbist reliability behavior of typical systems with two types of failure / K.Y. Cai, C.Y. Wen, M. L. Zhang // Fuzzy Sets and Systems. 1991. № 43. — С. 17 - 32.

175. Chang J.R., Chang K. H., Liao, S. H., and Cheng, C. H. . The reliability of general vague fault-tree analysis of weapon systems fault diagnosis // Soft Computing. 2006. Т. 10. — С. 531-542.

176. Chapman W.L. System design is an NP-complete problem / W. L. Chapman, J. Rozenblit, and A. T. Bahill // Systems Engineering, The Journal of INCOSE. 2001. № 4(3). — С. 222-229.

177. Chen S. An overview of quality of service routing for next-generation high-speed networks: problems and solutions / S. Chen, K. Nahrstedt // IEEE Networks 1998. № 12. — С. 64-79 p.

178. Chen S.M. Analyzing fuzzy system reliability using vague set theory / S. M. Chen // International Journal of Applied Science and Engineering. 2003. № 1. — С. 82 - 88.

179. Chen S.M. Arithmetic operations between vague sets / S.M. Chen // Proceedings of the International Joint Conference of CFSA/IFIS/SOFT’95 on Fuzzy Theory and Applications, Taipei, Taiwan, Re¬public of China. 1995. — С. 206 - 211.

180. Curtain R. Functional analysis in modern applied mathematics / R. Curtain, A. J. Pritchard : Academic Press, 1997.

181. Ein-Dor P. Grosh's law re-revisited: CPU power and the cost of computation // Communication of the ACM. 1985. Т. 28. № 2. — С. 142-151.

182. Franke E. Zur Aufwadsbewertung dezentraler bzw. zentraler Prozesrechnerstrukturen / Forschungsinstitut // AEG-TELEFUNKEN Ulm. 1986. — С. 146-159.

183. Gau W.L. Vague sets / W. L. Gau, D. J. Buehrer // IEEE Transactions on Systems, Man, and Cybernetics. 1993. № 23. — С. 610 - 614.

184. Gawlick R. Liveness in timed and untimed systems / R. Gawlick, R. Segala, J. F. Sogaard-Anderson, N. Lynch & B. Lampson // Technical report TR-587, MIT Computer Science Laboratory. 1993. — C. 289.

185. Gottlieb J. Prüfer numbers: A poor representation of spanning trees for evolutionary search / Jens Gottlieb, Bryant A. Julstrom, Günther R. Raidl, and Franz Rothlauf // Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-2001). 2001. — C. 343–350.

186. Gruber T.R. The role of common ontology in achieving sharable, reusable knowledge bases [Текст] / J.A. Allen, R. Fikes, T.R. Gruber, E. Sandewell – eds. Morgan Kaufmann // 1991. — C. 601-602.

187. Gumzej R. Real-time Systems' Quality of Service. Introducing Quality of Service Considerations in the Life-cycle of Real-time Systems / R. Gumzej, W. A. Halang. — London : Springer-Verlag London Limited, 2010. — 145 p.

188. Ionescu D. Designing supervisor for real-time systems. Theories and experiences for real-time system development / D. Ionescu. : World Scientific Publishing Company, 1994. — P. 103-128.

189. J.-B. Kruger. Bewertungsmasstabe fur den Zuverlassigkeitsvergleich von zentral und dezentral arbeitenden Prozesautomatisierungssystemen // Regelungstechnishe Praxis. 1978. Т. 20. № 8. — С. 225-252.

190. K. E. Knight. Changes in computer perfoirmance. A historical view // Datamation. 1966. № 9. — С. 40-54.

191. Kalman R. Mathematical theory of dynamic systems / P. Falb & M. A. Arbib. — New York : Academic Press, 1969.

192. Kleinrock L. Distributed systems. Special issue // Communication of the ACM. 1985. Т. 28. № 11. — С. 1200-1213.

193. Kosolapov A. Application of Network Information Technologies to Integration of Educational Process Control and Manage-ment in Technical University / O. Pshinko, B. Bodnar, A. Raspopov, A. Kosolapov // International Conference on Engineering Education, ICEE'99, Ostrava - Prague: VSB - Technical University of Ostrava. 1999. — С. 12.

194. Krogstie J. A. L. Opdahl, S. Brinkkemper Conceptual Modelling in Information Systems Engineering / J. Krogstie, A. L. Opdahl, S. Brinkkemper. — Berlin : Springer-Verlag Berlin Heidelberg, 2007. — 356 p.

195. Kumar A., Yadav, S. P., Kumar, S. Fuzzy System Reliability Using Different Types of Vague Sets // Int. Journal of Applied Science and Engineering. 2008. T. 6. № 1. — C. 71-83.

196. Landau A. A METHODOLOGICAL FRAMEWORK FOR BUSINESS-ORIENTED MODELING OF IT INFRASTRUCTURE / A. Landau, S. Wasserkrug, D. Gilat, IBM Haifa Research Lab University Campus Carmel Mountains Haifa, 31905, ISRAEL // Proceedings of the 2004 Winter Simulation Conference R .G. Ingalls, M. D. Rossetti, J. S. Smith, and B. A. Peters, eds. . 2004. — C. 1-9.

197. Leeman H., Dedene G. Grosch's law: a statistical illusion? / Onderzoeksrapport NR 9631 // Katholieke Universiteit Leuven. 1996. — C. 1-16.

198. Lin J.Y. Temporal Logic Approach to the Analysis and Synthesis of Discrete Event Systems. PhD thesis / J. Y. Lin. — University of Ottawa, 1993.

199. Liu T.S. The Role of a Maintenance Processor for a General-Purpose Computer System / T.S. Liu // IEEE Trans Comput. 1984. T. 33. — C. 507-517.

200. Lixia H. A Nove Genetic Algorithm for the Degree-constrained Minimum Spanning Tree Problem / Lixia Han // Computer Engineering and Applications. 2006. T. 42(31). — C. 37-42

201. Manna Z. The temporal logic of reactive and concurrent systems / Z. Manna & A. Pnueli. : Springer-Verlag, 1992. — 350 p.

202. Ostroff J.S. Temporal logic for real-time systems / J. S. Ostroff — NewYork : John Wiley and Sons, 1989.

203. Panzieri F. Системы реального времени: основные понятия [Електронний ресурс] / F. Panzieri, R. Davoli. Laboratory for Computer Science. University of Bologna. Режим доступу : ftp:// ftp.cs.unibo.it:/pub/TR/UBLCS // Техническое описание UBLCS-93-22. Октябрь. 1993.

204. René J. Chevance Server Architectures. Multiprocessors, Clusters, Parallel Systems, Web Servers, and Storage Solutions / J. René. — USA : Elsevier Digital Press, 2005. — 709 p.

205. Segala R. Modeling and Verification of Randomized Distributed Systems / R. Segala // PhD thesis, Massachusetts Institute of Technology. 1995. — С. 1-123.

206. Shan T.C. Towards a Systematic Method for Solutions Architecting. Handbook of research on modern systems analysis and design technologies and applications / Tony C. Shan, Winnie W. Hua. — N.Y. : IGI Global, 2009. — P. 1-22.

207. Singer D. A fuzzy set approach to fault tree and reliability analysis // Fuzzy Sets and Systems. 1990. Т. 34. № 2. — С. 145-155.

208. Sogaard-Anderson J.F. Correctness of communication protocols / J. F. Sogaard-Anderson, N. Lynch & B. Lampson // Technical report TR-589, MIT Computer Science Laboratory. 1993. — С. 4-17.

209. Stephen Blacketer S. ISO 9001:2001 Quality Management System - System Development Methodologies MBP5002. 22nd November 2001, вер. 1.0.2. / Stephen Blacketer —London : ISO, 2001. — 6p.

210. Streeter D.N. Centralization or dispersion of computing facilitien // IBM System Journal. 1973. № 3. — С. 283-301.

211. Topfer Y., Reinig G. Automatisierungsstrukturen, ihre Systematishe und Rechnergestutztg Gestaltung / Technische Universitat Dresden // Wissenschaftliche Berichte. 1986. Т. 35. № 1. — С. 130-138.

212. W. Fritzsch. Anlagenstrukturen und Steuerunskonzepte mit Mikroprozesrechnern // Messen-Steuern-Regeln. 1977. Т. 20. № 12. — С. 662-667.

213. Yzhak R. Distributed computing system as an altrnative to a central computing center. Cost effective analysis // 14 Conf. Elec. and Electron. Eng., Israel, Tel-Aviv, 26-28 March. 1986. — С. 1.2.4/1-1.2.4/6.

214. Zadeh L.A. Fuzzy sets // Inform. Control. 1965. Т. 8. № 3. — С. 338-353.

215. Zarecky S. The newest trends in marshalling yards automation / S. Zarecky, J. Grun, J. Zilka // Transport problem. 2008. Т. 3. № 4, Part 1. — С. 87-95.

216. Zhang R. Fuzzy Control of Queuing Systems / R. Zhang, Y.A. Phillis, V.S. Kouikoglou. — London : Springer-Verlag, 2005. — 175 p.